



**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**

BRNO UNIVERSITY OF TECHNOLOGY

**FAKULTA INFORMAČNÍCH TECHNOLOGIÍ**

FACULTY OF INFORMATION TECHNOLOGY

**ÚSTAV INTELIGENTNÍCH SYSTÉMŮ**

DEPARTMENT OF INTELLIGENT SYSTEMS

**HARDWAROVÁ REALIZACE NUMERICKÉHO  
INTEGRÁTORU S METODOU VYŠŠÍHO ŘÁDU**

HARDWARE REALIZATION OF HIGHER ORDER NUMERICAL INTEGRATOR

**DIPLOMOVÁ PRÁCE**

MASTER'S THESIS

**AUTOR PRÁCE**

AUTHOR

**Bc. FRANTIŠEK MATEČNÝ**

**VEDOUcí PRÁCE**

SUPERVISOR

**Ing. VÁCLAV ŠÁTEK, Ph.D.**

**BRNO 2018**

**Vysoké učení technické v Brně - Fakulta informačních technologií**

Ústav inteligentních systémů

Akademický rok 2017/2018

**Zadání diplomové práce**

Řešitel: **Matečný František, Bc.**

Obor: Informační systémy

Téma: **Hardwarová realizace numerického integrátoru s metodou vyššího řádu**  
**Hardware Realization of Higher Order Numerical Integrator**

Kategorie: Modelování a simulace

**Pokyny:**

1. Seznamte se s numerickými metodami pro řešení obyčejných diferenciálních rovnic (počátečních úloh). Zaměřte se na metody vyšších řádů využívající metodu Taylorovy řady.
2. Navrhněte dva typy vícevstupových integrátorů, jeden podporující operaci násobení a druhý operaci dělení. Integrátory umožňují operace s pevnou a pohyblivou řádovou čárkou.
3. Implementujte v provedení FPGA elementární procesor využívající navržené integrátory. Procesor bude implementovat funkcionality komplexní spojité simulace.
4. Ověřte činnost navrženého procesoru s integrátory na vhodných příkladech a určete dosažené zrychlení vašeho integrátoru ve srovnání s klasickými metodami nižšího řádu (metody Euler a Runge-Kutta 2. a 4. řádu).

**Literatura:**

- M. Kraus: Paralelní výpočetní architektury založené na numerické integraci, Disertační práce, FIT VUT v Brně, 2013.
- Kunovský, J.: Modern Taylor series method. Habilitation work, VUT Brno, 1994.
- M. Kubíček, M. Dubcová, D. Janovská: Numerické metody a algoritmy. VŠCHT Praha, 2005. ISBN 80-7080-558-7
- J. P. Deschamps, G. D. Sutter, E. Cantó: Guide to FPGA Implementation of Arithmetic Functions, Springer, 2012.
- další dle pokynů vedoucího

Při obhajobě semestrální části projektu je požadováno:

- Body 1 a 2.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese

<http://www.fit.vutbr.cz/info/szz/>

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci dřívějších projektů (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Šátek Václav, Ing., Ph.D., UITS FIT VUT**

Datum zadání: 1. listopadu 2017

Datum odevzdání: 23. května 2018

**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**  
Fakulta informačních technologií  
Ústav inteligentních systémů  
602 00 Brno, Božetěchova 2

doc. Dr. Ing. Petr Hanáček  
vedoucí ústavu

## Abstrakt

Práca popisuje numerickú integráciu a riešenie obyčajných diferenciálnych rovníc pomocou metódy Taylorovej rady v rôznych typoch integrátorov. Ďalej je popísaná aritmetika pevnej a pohyblivej rádovej čiarky. Následne sú predstavené návrhy a spôsob výpočtu paralelných integrátorov s operáciou násobenia a delania v prevedení pevnej a pohyblivej rádovej čiarky. Súčasťou práce je taktiež popísanie integrátorov vo VHDL a ich implementácia v FPGA. V závere práce je analyzovaná časová zložitosť výpočtu s inými numerickými metódami.

## Abstract

This work describes numerical integration and solution for ordinary differential equations by the Taylor series by different types of integrators. The next part is a description of floating point and fixed point arithmetic. Subsequently, we are presenting designs and calculation methods for parallel multiplication and division integrators in floating point and fixed point arithmetic. The designs were realized in VHDL and implemented on FPGA. Finally we summarize the proposed solution and compare time complexity with another numerical methods.

## Kľúčové slová

diferenciálna rovnica, numerická integrácia, Taylorova rada, pevná rádová čiarka, pohyblivá rádová čiarka, integrátor, FPGA

## Keywords

differential equation, numeric integration, Taylor series, fixed point, floating point, integrator, FPGA

## Citácia

MATEČNÝ, František. *Hardwarová realizace numerického integrátoru s metodou vyššího řádu*. Brno, 2018. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Václav Šátek, Ph.D.

# Hardwarová realizace numerického integrátoru s metodou vyššího řádu

## Prehlásenie

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne pod vedením pána Ing. Václava Šátka, Ph.D. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....

František Matečný

22. mája 2018

## Podakovanie

Chcel by som sa poďakovať svojmu školiteľovi Ing. Václavovi Šátkovi, Ph.D., za odborné vedenie práce, podporu, vecné pripomienky, komentáre a rady k mojej práci. Moje poďakovanie taktiež patrí mojej rodine, priateľke Katke, kamarátom a všetkým ostatným, ktorí ma podporovali počas celej doby môjho štúdia a bez ktorých by táto práca nemohla vzniknúť.

# Obsah

<b>1</b>	<b>Úvod</b>	<b>4</b>
<b>2</b>	<b>Numerická integrácia</b>	<b>5</b>
2.1	Taylorova rada . . . . .	6
2.2	Eulerova metóda . . . . .	6
2.3	Runge-Kutta . . . . .	6
2.3.1	Runge-Kutta 2. rádu . . . . .	7
2.3.2	Runge-Kutta 4. rádu . . . . .	7
<b>3</b>	<b>Riešenie diferenciálnych rovníc Taylorovou radou</b>	<b>8</b>
3.1	Riešenie diferenciálnej rovnice s operáciou násobenia . . . . .	9
3.2	Riešenie diferenciálnej rovnice s operáciou delenia . . . . .	10
<b>4</b>	<b>Reprezentácia operandov</b>	<b>12</b>
4.1	Pevná rádová čiarka . . . . .	12
4.2	Pohyblivá rádová čiarka . . . . .	13
4.2.1	Súčet a rozdiel . . . . .	15
4.2.2	Násobenie a delenie . . . . .	16
<b>5</b>	<b>Numerické integrátory</b>	<b>17</b>
5.1	Návrh násobiaceho integrátora v pevnej rádovej čiarke . . . . .	18
5.2	Návrh deliaceho integrátora v pevnej rádovej čiarke . . . . .	19
5.3	Návrh jednovstupového integrátora . . . . .	21
5.4	Návrh komponentu na spracovanie exponentov . . . . .	22
5.5	Návrh násobiacieho integrátora v pohyblivej rádovej čiarke . . . . .	23
5.6	Návrh deliaceho integrátora v pohyblivej rádovej čiarke . . . . .	24
5.7	Sústava diferenciálnych rovníc . . . . .	26
<b>6</b>	<b>Popísanie integrátorov vo VHDL</b>	<b>28</b>
6.1	Integrátory v pevnej rádovej čiarke . . . . .	29
6.2	Integrátory v pohyblivej rádovej čiarke . . . . .	35
6.3	Sústava integrátorov . . . . .	36
<b>7</b>	<b>Analýza</b>	<b>39</b>
<b>8</b>	<b>Záver</b>	<b>44</b>
	<b>Literatúra</b>	<b>45</b>



# Zoznam obrázkov

4.1	Rôzne formáty fixed point aritmetiky [2]. . . . .	12
4.2	IEEE 754 formát s jednoduchou presnosťou. . . . .	13
4.3	IEEE 754 formát s dvojitou presnosťou. . . . .	13
5.1	Schéma jednovstupového a dvojevstupového integrátora. . . . .	17
5.2	Paralelno-paralelný násobiaci integrátor [14]. . . . .	18
5.3	Paralelno-paralelný deliaci integrátor [4]. . . . .	20
5.4	Paralelný jednovstupový integrátor v pohyblivej rádovej čiarke. . . . .	21
5.5	EXP - blok pracujúci s exponentmi. . . . .	22
5.6	Paralelno-paralelný násobiaci integrátor v pohyblivej rádovej čiarke. . . . .	24
5.7	Paralelno-paralelný deliaci integrátor v pohyblivej rádovej čiarke. . . . .	25
5.8	Schéma zapojenia integrátorov [4]. . . . .	26
6.1	FX aritmetika numerického integrátora. . . . .	29
6.2	FX aritmetika deliaceho integrátora na 32 bitoch. . . . .	30
6.3	FX aritmetika deliaceho integrátora na 64 bitoch. . . . .	30
6.4	Schéma zapojenia integrátorov. . . . .	37
6.5	Prípravky FitKit 2.0 a VIRTEX-5, s ktorými sa pracovalo. . . . .	38
7.1	Stavy jednotlivých integrátorov. . . . .	39
7.2	Schéma zapojenia integrátorov. . . . .	41
7.3	Simulácia sústavy rovníc 6.2 Taylorovou radou 8. rádu s $h = 1$ a $h_{max} = 1$ . . . . .	43
7.4	Simulácia sústavy rovníc 6.2 Taylorovou radou 8. rádu s $h = 0.5$ a $h_{max} = 1$ . . . . .	43

# Zoznam tabuliek

4.1	Štandard IEEE 754 [11]. . . . .	14
4.2	Výsledok operácie sčítania so špeciálnymi hodnotami. . . . .	15
4.3	Výsledok operácie násobenia so špeciálnymi hodnotami. . . . .	16
4.4	Výsledok operácie delenia so špeciálnymi hodnotami. . . . .	16
5.1	Význam skratiek použitých v obrázku 5.2. . . . .	19
5.2	Význam skratiek v obrázku 5.3. . . . .	20
5.3	Význam skratiek použitých v obrázku 5.5. . . . .	22
6.1	Kontrolór násobiaceho integrátora v FX. . . . .	32
6.2	Kontrolór násobiaceho integrátora v FP. . . . .	32
6.3	Kontrolór deliaceho integrátora v FX. . . . .	33
6.4	Kontrolór deliaceho integrátora v FP. . . . .	34
6.5	Kontrolór jednovstupového integrátora v FP. . . . .	35
6.6	Kontrolór jednovstupového integrátora v FP v zapojení do sústavy. . . . .	36
7.1	Porovnanie presnosti a rýchlosti numerických metód. . . . .	42
7.2	Porovnanie presnosti a rýchlosti numerických metód. . . . .	42



# Kapitola 1

## Úvod

Hlavným cieľom tejto práce je návrh hardvérových komponentov na riešenie rozsiahlych diferenciálnych rovníc. Diferenciálne rovnice sa väčšinou riešia pomocou numerickej integrácie, a teda s použitím vhodných numerických metód. Hardvérový komponent využívajúci numerickú integráciu sa nazýva numerický integrátor.

V kapitole 2 sú predstavené rôzne numerické metódy - Eulerova metóda, metóda Runge-Kutta a metóda Taylorovej rady. Najväčšia pozornosť je venovaná metóde Taylorovej rady, ktorá poskytuje vhodný pomer medzi rýchlosťou a presnosťou [3]. Bližší popis a práca s touto metódou sú uvedené v kapitole 3. Ukážeme si rozdelenie Taylorovej rady na jednotlivé členy a následne úpravu týchto členov tak, aby bolo možné výpočet čo najviac paralelizovať a optimalizovať. Táto úprava bude realizovaná na obyčajných diferenciálnych rovniciach s operáciou násobenia a delenia. Takto upravené a vytvorené rovnice budú následne použité pri návrhu rôznych typov numerických integrátorov.

Kapitola 4 sa zaoberá reprezentáciou operandov v pevnej a v pohyblivej rádovej čiarkke. Pri použití pohyblivej rádovej čiarky sú uvedené postupy výpočtu znamienka, exponentu a mantisy na jednoduchých matematických operáciách ako sú sčítanie, odčítanie, násobenie a delenie.

V kapitole 5 sú predstavené a popísané návrhy jednotlivých integrátorov a popis ich činnosti. Podľa rovníc uvedených v kapitole 3 sú navrhnuté paralelné numerické integrátory s operáciou násobenia a delenia. Oba integrátory sú navrhnuté v prevedení pevnej a pohyblivej rádovej čiarky. Vzájomným zapojením navrhnutých numerických integrátorov je možné riešiť rozsiahle diferenciálne rovnice.

Navrhnuté integrátory budú popísane vo VHDL. Podrobnejší popis implementácie obsahuje kapitola 6. Integrátory následne otestujeme v prostredí *ModelSim* použitím *tesbanchu*. Takto pripravené integrátory sú implementované na FPGA, a to samostatne a v zapojení do sústavy.

Kapitola 7 obsahuje časovú analýzu jednotlivých integrátorov v zapojení do sústavy. Ďalej je analyzovaná presnosť použitej Taylorovej rady s metódami nižšieho rádu, ktoré sú popísane v kapitole 2.

## Kapitola 2

# Numerická integrácia

Diferenciálne rovnice sú matematické rovnice, v ktorých ako premenné vystupujú derivácie funkcií. Najvyššia derivácia v rovnici udáva rád rovnice. Rovnice, ktoré obsahujú derivácie len podľa jednej premennej, sa nazývajú obyčajné diferenciálne rovnice (ODR). Rovnice, ktoré obsahujú derivácie podľa viacerých premenných, sú takzvané parciálne diferenciálne rovnice (PDR). V tejto práci sa budeme zaoberať s diferenciálnymi rovnicami prvého rádu. Tie je možné vyjadriť v tvare

$$y'(t) = f(t, y(t)), \quad (2.1)$$

kde je potrebné zvoliť hodnotu funkcie v bode  $t = t_0$ , t. j. počiatočnú podmienku

$$y(t_0) = y_0. \quad (2.2)$$

Obe rovnice (2.1) a (2.2) tvoria počiatočnú úlohu, taktiež nazývanú ako Cauchyova úloha. Budeme uvažovať, že funkcia (2.1) je Lipschitzovsky spojitá a ohraničená, potom Cauchyova úloha má práve jedno riešenie [16]. Diferenciálnu rovnicu je možné riešiť analyticky alebo použitím numerickej metódy. Pri väčšine praktických úloh je analytické riešenie veľmi zložitá, preto sa používa skôr riešenie numerické. Základným princípom numerického riešenia je diskretizácia premenných, keď spojitú veličinu nahradíme postupnosťou diskretných bodov. Pri použití dostatočne hustého rozloženia bodov môžeme približne reprezentovať spojitú veličinu. Vzdialenosť medzi dvoma susednými bodmi sa nazýva krok metódy. Numerické metódy pri svojom výpočte používajú niekoľko predchádzajúcich krokov. Podľa počtu týchto krokov rozdeľujeme numerické metódy na metódy jednokrokové a viackrokové. Jednokrokové metódy pri svojom výpočte používajú len jeden predchádzajúci krok, viackrokové metódy využívajú niekoľko predchádzajúcich krokov. Pri numerických metódach je teda potrebné zvoliť počiatočný stav, t. j. počiatočnú podmienku riešenej úlohy. Od nej sa následne počíta hodnota funkcie v ďalšom bode [5].

Hlavnými kritériami pri numerických metódach sú presnosť a rýchlosť. Tie je možné ovplyvniť veľkosťou integračného kroku a rádom integračnej metódy. Pri počítaní numerickými metódami nedostávame teoreticky presné riešenie, ale výsledok konverguje k správne riešeniu, a teda dostávame výsledok s určitou presnosťou. Výsledná chyba výpočtu je súčet lokálnej a akumulovanej chyby. Lokálna chyba zahŕňa chybu numerickej metódy a zaokrúhľovaciu chybu, ktorá môže byť spôsobená typom hardvérovej architektúry, ako napríklad použitím pevnej alebo pohyblivej rádovej čiarky, ktoré sú bližšie popísané v kapitole 4. Akumulovaná chyba je súčtom lokálnych chýb, čiže sa počas výpočtu zvyšuje. Chyba v jednom kroku teda ovplyvňuje výsledky krokov nasledujúcich.

## 2.1 Taylorova rada

Táto numerická metóda je tvorená nekonečným radom, avšak na výpočet sa používa len niekoľko jej členov. Počet použitých členov udáva rád metódy. Čím väčší počet členov použijeme, tým je výsledok presnejší. Počet použitých členov môže byť zadaný fixne, alebo sa môže dynamicky meniť v závislosti od požadovanej presnosti. Presnosť sa počíta pomocou viacerých najvyšších členov. Po dosiahnutí požadovanej presnosti výpočet končí. Nekonečnú Taylorovu radu môžeme zapísať:

$$y_{i+1} = y_i + hy'_i + \frac{h^2}{2!}y''_i + \frac{h^3}{3!}y'''_i + \frac{h^4}{4!}y^{(4)}_i + \dots + \frac{h^n}{n!}y^{(n)}_i, \quad (2.3)$$

kde  $h$  je veľkosť integračného kroku a  $i$  označuje krok diskretizovanej veličiny. Ďalšie popísané metódy sú odvodené od Taylorovej rady.

## 2.2 Eulerova metóda

Najjednoduchšou jednokrokovou metódou na riešenie počiatkovej (Cauchyho) úlohy je Eulerova metóda. Je to Taylorova metóda 1. rádu, keďže používa len prvé dva členy Taylorovej rady. Je preto rýchla, no menej presná. Zapisuje sa nasledovne:

$$y_{i+1} = y_i + hy'_i. \quad (2.4)$$

Zvolením dostatočne malého integračného kroku  $h$  môžeme zvýšiť jej presnosť. Čím je však krok menší, tým je výpočet pomalší.

## 2.3 Runge-Kutta

Ďalšou veľmi známou numerickou metódou je Runge-Kutta. Všeobecná schéma tejto metódy má tvar

$$\begin{aligned} y_{i+1} &= y_i + \sum_{j=1}^r \alpha_j k_j \\ k_1 &= f(x_i, y_i) \\ k_j &= f(x_i + \lambda_j h, y_i + \mu_j h k_{j-1}), \quad j = 2, \dots, r, \end{aligned} \quad (2.5)$$

kde  $\alpha_j$ ,  $\lambda_j$  a  $\mu_j$  sú vhodne zvolené konštanty a  $r$  určuje rád metódy. Ako je v uvedených rovniciach vidieť, pri výpočte sa používajú medzivýpočty  $k$ , ktorých počet sa rovná rádu metódy. Najznámejšie a najčastejšie používané varianty sú Runge-Kutta 2. a 4. rádu, ktoré sú popísané nižšie.

### 2.3.1 Runge-Kutta 2. rádu

Táto metóda je oproti Eulerovej metóde presnejšia, ale pri rovnakej veľkosti integračného kroku vyžaduje viac operácií. Na výpočet používa dva medzivýpočty  $k_1$  a  $k_2$ . Má nasledujúci tvar:

$$\begin{aligned}y_{i+1} &= y_i + \frac{1}{2}h(k_1 + k_2) \\k_1 &= f(t_i, y_i) \\k_2 &= f(t_{i+1}, y_i + hk_1).\end{aligned}\tag{2.6}$$

### 2.3.2 Runge-Kutta 4. rádu

Runge-Kutta 4. rádu je najpoužívanější tvar tejto metódy. Môžeme ho zapísať nasledovne:

$$\begin{aligned}y_{i+1} &= y_i + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4) \\k_1 &= f(t_i, y_i) \\k_2 &= f(t_{i+1} + \frac{1}{2}h, y_i + \frac{1}{2}hk_1) \\k_3 &= f(t_{i+1} + \frac{1}{2}h, y_i + \frac{1}{2}hk_2) \\k_4 &= f(t_{i+1} + h, y_i + hk_3).\end{aligned}\tag{2.7}$$

Napriek tomu, že sú pri výpočte použité štyri medzivýpočty, má táto metóda dobrý pomer rýchlosti a presnosti.

## Kapitola 3

# Riešenie diferenciálnych rovníc Taylorovou radou

Na riešenie diferenciálnych rovníc je možné upraviť základný tvar Taylorovej rady (2.3) tak, aby sa dali jednotlivé operácie vykonávať paralelne. Prevod jednoduchej obyčajnej diferenciálnej rovnice je prevzatý z [4]. Ďalšími možnými zdrojmi sú [7], [2]. V ďalších častiach tejto práce budeme používať označenie  $N$  pre použitý rád Taylorovej metódy, t. j. maximálny počet použitých členov. Symbol  $n$  bude označovať aktuálne spracovávaný člen Taylorovej rady.

Majme zadanú konkrétnu obyčajnú diferenciálnu rovnicu (počiatočnú úlohu):

$$y' = y, \quad y(0) = y_0. \quad (3.1)$$

Z tohoto vzťahu vyplýva, že

$$y = y' = y'' = y''' = y^{(4)} = \dots = y^{(N)}. \quad (3.2)$$

Po dosadení do Taylorovej rady (2.3) získame

$$y_{i+1} = y_i + hy_i + \frac{h^2}{2!}y_i + \frac{h^3}{3!}y_i + \frac{h^4}{4!}y_i + \dots + \frac{h^N}{N!}y_i. \quad (3.3)$$

To je možné prepísať na

$$y_{i+1} = y_i + DY1_i + DY2_i + DY3_i + DY4_i + \dots + DY(N)_i, \quad (3.4)$$

kde je význam jednotlivých členov nasledujúci:

$$\begin{aligned} DY1_i &= hy_i \\ DY2_i &= \frac{h^2}{2!}y_i = \frac{h}{2}DY1_i \\ DY3_i &= \frac{h^3}{3!}y_i = \frac{h}{3}DY2_i \\ DY4_i &= \frac{h^4}{4!}y_i = \frac{h}{4}DY3_i \\ &\vdots \end{aligned} \quad (3.5)$$

Všeobecný zápis:

$$DY(n)_i = \frac{h^n}{n!}y_i = \frac{h}{n}DY(n-1)_i$$

Na základe týchto vzťahov je možné riešiť jednoduché diferenciálne rovnice, ako to je uvedené napríklad v [6] alebo v [2]. Zo vzťahov taktiež vyplýva, že každý ďalší člen Taylorovej rady je počítaný z predchádzajúceho člena, čo vedie k zefektívneniu výpočtu, a to najmä pri vyšších deriváciách, v ktorých je výpočet častokrát zložitý. Podobným postupom je možné upraviť diferenciálne rovnice, ktoré obsahujú operáciu násobenia alebo delenia, a tiež tak zvýšiť efektivitu výpočtu týchto výpočtovo náročnejších operácií.

### 3.1 Riešenie diferenciálnej rovnice s operáciou násobenia

Všeobecný zápis pre diferenciálnu rovnicu s operáciou násobenia je nasledujúci:

$$y' = qr, \quad y(0) = y_0. \quad (3.6)$$

Pre výpočet každej premennej v rovnici je potrebné použiť Taylorovu radu v tvare (3.4). Jednotlivé Taylorove rady vyzerajú nasledovne:

$$y_{i+1} = y_i + DY1_i + DY2_i + DY3_i + DY4_i + \dots + DY(N)_i \quad (3.7)$$

$$q_{i+1} = q_i + DQ1_i + DQ2_i + DQ3_i + DQ4_i + \dots + DQ(N)_i \quad (3.8)$$

$$r_{i+1} = r_i + DR1_i + DR2_i + DR3_i + DR4_i + \dots + DR(N)_i. \quad (3.9)$$

Spočítame jednotlivé derivácie rovnice (3.6):

$$\begin{aligned} y' &= qr \\ y'' &= q'r + qr' \\ y''' &= q''r + 2q'r' + qr'' \\ y^{(4)} &= q'''r + 3q''r' + 3q'r'' + qr''' \\ &\vdots \end{aligned}$$

Ako je vidieť, derivácie vytvárajú Pascalov trojuholník a môžeme ich všeobecne zapísať ako

$$y^{(n+1)} = \sum_{k=0}^n \binom{n}{k} q^{(n-k)} r^{(k)}.$$

Z derivácií odvodíme jednotlivé členy Taylorovej rady (2.3), ktoré majú nasledujúci význam:

$$\begin{aligned} \frac{DY1_i}{h} &= DQ0_i DR0_i \\ \frac{DY2_i}{\frac{h^2}{2!}} &= \frac{DQ1_i}{h} DR0_i + DQ0_i \frac{DR1_i}{h} \\ \frac{DY3_i}{\frac{h^3}{3!}} &= \frac{DQ2_i}{\frac{h^2}{2!}} DR0_i + 2 \frac{DQ1_i}{h} \frac{DR1_i}{h} + DQ0_i \frac{DR2_i}{\frac{h^2}{2!}} \\ \frac{DY4_i}{\frac{h^4}{4!}} &= \frac{DQ3_i}{\frac{h^3}{3!}} DR0_i + 3 \frac{DQ2_i}{\frac{h^2}{2!}} \frac{DR1_i}{h} + 3 \frac{DQ1_i}{h} \frac{DR2_i}{\frac{h^2}{2!}} + DQ0_i \frac{DR3_i}{\frac{h^3}{3!}} \\ &\vdots \end{aligned} \quad (3.10)$$

Po úprave dostaneme konečný tvar jednotlivých členov:

$$DY1_i = hq_i r_i \quad (3.11)$$

$$DY2_i = \frac{h}{2}(DQ1_i DR0_i + DQ0_i DR1_i) \quad (3.12)$$

$$DY3_i = \frac{h}{3}(DQ2_i DR0_i + DQ1_i DR1_i + DQ0_i DR2_i) \quad (3.13)$$

$$DY4_i = \frac{h}{4}(DQ3_i DR0_i + DQ2_i DR1_i + DQ1_i DR2_i + DQ0_i DR3_i) \quad (3.14)$$

$\vdots$

Všeobecný zápis:

$$DY(n)_i = \frac{h}{n} \cdot \left( \sum_{k=1}^n DQ(n-k)_i \cdot DR(k-1)_i \right). \quad (3.15)$$

Uvedený prevod vychádza z práce [14]. Jednotlivé rovnice členov Taylorovej rady budú slúžiť pri návrhu a pri implementácii násobiacich integrátorov.

## 3.2 Riešenie diferenciálnej rovnice s operáciou delenia

Podobne ako v predchádzajúcom prípade, zapíšme diferenciálnu rovnicu s operáciou delenia nasledovne:

$$y' = \frac{u}{v}, \quad y(0) = y_0. \quad (3.16)$$

Jednotlivé Taylorove rady pre premenné v rovnici vyzerajú nasledovne:

$$y_{i+1} = y_i + DY1_i + DY2_i + DY3_i + DY4_i + \dots + DY(N)_i \quad (3.17)$$

$$u_{i+1} = u_i + DU1_i + DU2_i + DU3_i + DU4_i + \dots + DU(N)_i \quad (3.18)$$

$$v_{i+1} = v_i + DV1_i + DV2_i + DV3_i + DV4_i + \dots + DV(N)_i. \quad (3.19)$$

Ďalšie derivácie rovnice (3.16) sú:

$$\begin{aligned} y'' &= \frac{u'v - uv'}{v^2} = \frac{1}{v}(u' - y'v') \\ y''' &= \left( \frac{1}{v}(u' - y'v') \right)' = \frac{1}{v}(u'' - 2y''v' - y'v'') \\ y^{(4)} &= \left( \frac{1}{v}(u'' - 2y''v' - y'v'') \right)' = \frac{1}{v}(u''' - 3y'''v' - 3y''v'' - y'v''') \\ &\vdots \end{aligned} \quad (3.20)$$

Aj tu je vidieť, že jednotlivé derivácie tvoria Pascalov trojuholník. Všeobecný zápis je nasledovný:

$$y^{(n+1)} = \frac{1}{v} \left( u^{(n)} - \left( \sum_{k=1}^n \binom{n}{k} y^{(n-k+1)} v^{(k)} \right) \right).$$

Po dosadení jednotlivých členov  $DV(n)_i$  a  $DU(n)_i$  do derivácií (3.20) dostaneme

$$\begin{aligned}
\frac{DY1_i}{h} &= \frac{1}{v}u \\
\frac{DY2_i}{\frac{h^2}{2!}} &= \frac{1}{v}\left(\frac{DU1_i}{h} - \frac{DY1_i}{h}\frac{DV1_i}{h}\right) \\
\frac{DY3_i}{\frac{h^3}{3!}} &= \frac{1}{v}\left(\frac{DU2_i}{\frac{h^2}{2!}} - 2\frac{DY2_i}{\frac{h^2}{2!}}\frac{DV1_i}{h} - \frac{DY1_i}{h}\frac{DV2_i}{\frac{h^2}{2!}}\right) \\
\frac{DY4_i}{\frac{h^4}{4!}} &= \frac{1}{v}\left(\frac{DU3_i}{\frac{h^3}{3!}} - 3\frac{DY3_i}{\frac{h^3}{3!}}\frac{DV1_i}{h} - 3\frac{DY2_i}{\frac{h^2}{2!}}\frac{DV2_i}{\frac{h^2}{2!}} - \frac{DY1_i}{h}\frac{DV3_i}{\frac{h^3}{3!}}\right) \\
&\vdots
\end{aligned} \tag{3.21}$$

Po úprave majú jednotlivé členy Taylorovej rady nasledovný tvar:

$$DY1_i = \frac{1}{v_i}(hu_i) \tag{3.22}$$

$$DY2_i = \frac{1}{2v_i}(DU1_i h - DY1_i DV1_i) \tag{3.23}$$

$$DY3_i = \frac{1}{3v_i}(DU2_i h - 2DY2_i DV1_i - DY1_i DV2_i) \tag{3.24}$$

$$DY4_i = \frac{1}{4v_i}(DU3_i h - 3DY3_i DV1_i - 2DY2_i DV2_i - DY1_i DV3_i) \tag{3.25}$$

$\vdots$

Všeobecný zápis:

$$DY(n)_i = \frac{1}{nDV0_i} \cdot \left( DU(n-1)_i \cdot h - \left( \sum_{k=1}^{n-1} (n-k) \cdot DY(n-k)_i \cdot DV(k)_i \right) \right). \tag{3.26}$$

Vyjadrenie jednotlivých členov Taylorovej rady vychádza z bakalárskej práce [4]. Takto upravené členy sú následne použité na návrh jednotlivých typov integrátorov popísaných v kapitole 5.



## Kapitola 4

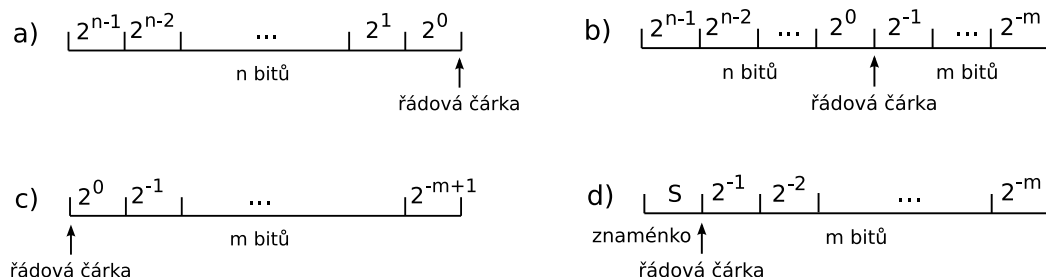
# Reprezentácia operandov

Čísla vo výpočtových systémoch sú reprezentované rôznymi spôsobmi v závislosti od zvolenej aritmetiky. Najpoužívanjšie reprezentácie čísiel sú zapísané v pevnej a v pohyblivej rádovej čiarkke (v angl. fixed point (FX) a floating point (FP)). Obe aritmetiky sú popísané v nasledujúcich podkapitolách. Informácie v tejto kapitole sú čerpané z [10], [11], [2], [8], [9].

### 4.1 Pevná rádová čiarka

Pri počítaní v pevnej rádovej čiarkke sú čísla reprezentované na  $k$  bitoch v tvare  $n.m$ , kde prvých  $n$  bitov tvorí časť čísla pred desatinnou čiarkou a zostávajúcich  $m$  bitov tvorí číslo za desatinnou čiarkou. Pozícia desatinnej čiarky je dopredu známa. V závislosti od jej pozície sa používajú rôzne formáty pevnej rádovej čiarky. Tie najpoužívanjšie sú znázornené na obrázku 4.1.

Obrázok 4.1a zobrazuje aritmetiku s nulovým počtom bitov za desatinnou čiarkou, a teda ide o celočíselnú aritmetiku na  $n$  bitoch. Vedľajší obrázok 4.1b zobrazuje aritmetiku s  $n$  bitmi pred desatinnou čiarkou a s  $m$  bitmi za desatinnou čiarkou. Ďalšia časť obrázku 4.1c je opakom 4.1a, kde je číslo reprezentované len za desatinnou čiarkou na  $m$  bitoch. Posledný obrázok 4.1d je podobný ako 4.1c, no má pridaný znamienkový bit  $S$ . V tejto práci budeme používať FX aritmetiku v tvare 4.1b s pridaným znamienkovým bitom na pozícii MSB (most significant bit).



Obr. 4.1: Rôzne formáty fixed point aritmetiky [2].

Vo fixed point aritmetike sa používajú rôzne kódy, napr. priamy kód, doplnkový kód či inverzný kód. V nasledujúcich kapitolách a pri návrhu integrátorov v pevnej rádovej čiarkke budeme používať doplnkový kód.

## 4.2 Pohyblivá rádová čiarka

Čísla uložené v pohyblivej rádovej čiarkke sú tvorené exponentom a mantisou. Všeobecný tvar na získanie hodnoty uloženej vo FP je nasledujúci:

$$X = B^E \cdot M \quad (4.1)$$

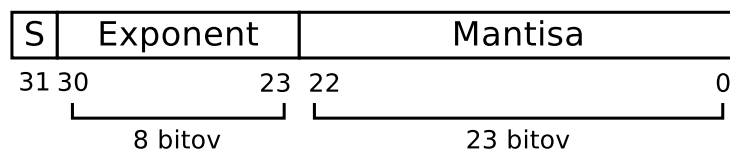
$X$  - výsledná hodnota

$B$  - základ sústavy

$E$  - hodnota exponenta

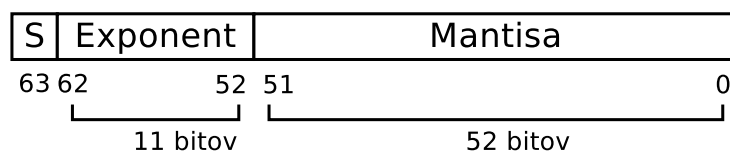
$M$  - mantisa

Zvýšením počtu bitov v exponente  $E$  sa zvýši rozsah hodnôt, ktorý je možné reprezentovať, a zvýšením počtu bitov v mantise  $M$  sa zvýši presnosť uložených čísel. Existuje veľa formátov uloženia čísel v pohyblivej rádovej čiarkke. Najpoužívanější a najrozšírenejší je štandard **IEEE 754**. Ten definuje vlastný formát uloženia čísel a viaceré formáty s rôznou presnosťou. Najpoužívanějšíe z nich sú formáty čísel s jednoduchou (single) a s dvojitou (double) presnosťou. Čísla s jednoduchou presnosťou sú uložené na 32 bitoch, kde MSB je znamienkový bit  $S$ , ďalších 8 bitov tvorí exponent  $E$  a zvyšných 23 bitov tvorí mantisu  $M$ .



Obr. 4.2: IEEE 754 formát s jednoduchou presnosťou.

Čísla s dvojitou presnosťou sú uložené v rovnakom formáte ako čísla s jednoduchou presnosťou, avšak počet bitov je zvýšený o dvojnásobok. Dvojitá presnosť je teda tvorená 64 bitmi, kde MSB je znamienkový bit  $S$ , ďalších 11 bitov tvorí exponent  $E$  a zvyšných 52 bitov tvorí mantisu  $M$ .



Obr. 4.3: IEEE 754 formát s dvojitou presnosťou.

Znamienko  $S$  nadobúda hodnotu 0, čo znamená kladné číslo; alebo hodnotu 1, čo znamená záporné číslo. Exponent je uložený v kóde s nepárnym posunutím o hodnotu BIAS. Táto hodnota je zvolená tak, aby bol uložený exponent vždy kladný. Pri jednoduchej presnosti má teda BIAS hodnotu 127 a pri dvojitej presnosti hodnotu 2047.

Hodnota mantisy je uložená v priamom kóde bez znamienka, znížená o hodnotu 1, keďže je tu použitá normalizácia. Mantisa je normalizovaná do tvaru  $1.M$ , kde sa jednotka neukladá – je skrytá, čím sa ušetrí jeden bit. Hodnotu takto uloženého čísla získame zo vzťahu

$$X_{754} = -1^S \cdot 2^{E-BIAS} \cdot (1, M) \quad (4.2)$$

$X_{754}$  - výsledná hodnota

$BIAS$  - 127 alebo 2047

$E$  - hodnota exponentu

$M$  - mantisa

Štandard IEEE 754 definuje aj špeciálne hodnoty ako kladnú/zápornú nulu, kladné/záporné nekonečno, či hodnotu NaN (not a number). Tieto hodnoty sú uvedené v tabuľke 4.1. Hodnota mantisy v normalizovanom tvare je v intervale  $< 1, 0; 2, 0$ ). Ak tomu tak nie je, ide o tzv. denormalizované číslo, a hodnota exponenta je braná ako  $-126$ . Štandard IEEE 754 definuje aj spôsob vykonávania základných matematických operácií, ktoré sú popísané v sekciách 4.2.1 a 4.2.2.

S (znamienko)	E (exponent)	M (mantisa)	význam
0/1	00000000	nulová hodnota	+/- 0
0/1	00000000	nenulová hodnota	+/- denormalizované číslo
0/1	1 - 254	ľubovoľná hodnota	+/- FP číslo
0/1	11111111	nulová hodnota	+/- $\infty$
0/1	11111111	nenulová hodnota	NaN

Tabuľka 4.1: Štandard IEEE 754 [11].

### 4.2.1 Súčet a rozdiel

Súčet a rozdiel v pohyblivej rádovej čiarke sa počíta podľa vzorcov

$$X + Y = (M_X \cdot 2^{E_X - E_Y} + M_Y) \cdot 2^{E_Y}, \text{ kde } E_X \leq E_Y \quad (4.3)$$

$$X - Y = (M_X \cdot 2^{E_X - E_Y} - M_Y) \cdot 2^{E_Y}, \text{ kde } E_X \leq E_Y. \quad (4.4)$$

Postup výpočtu operácie súčtu alebo rozdielu v pohyblivej rádovej čiarke podľa štandardu IEEE 754 ([12], [15]) je nasledovný:

1. Na začiatku výpočtu sa obe čísla skontrolujú na výskyt špeciálnych hodnôt z tabuľky 4.1. Ak ide o špeciálne číslo, výsledok sa určí podľa tabuľky 4.2. Inak sa pokračuje nasledujúcim bodom.
2. Vykoná sa porovnanie exponentov. Ak sú exponenty rozdielne, mantisu menšieho čísla posunieme o rozdiel exponentov doprava. Tým docielime rovnosť oboch exponentov. Pri posune je dôležité, aby sme nezabudli na jednotku, ktorá je skrytá kvôli normalizácii. Posun sa vykonáva spolu s toutou jednotkou.
3. Následne sa porovnajú znamienka a podľa výsledku sa vykoná súčet alebo rozdiel mantisy väčšieho čísla a posunutej mantisy. Ak došlo k pretečeniu, mantisa výsledku sa posunie o jeden bit doprava a hodnota exponentu sa zvýši. Aby bolo možné pretečenie detekovať, je potrebné vykonávať sčítanie alebo odčítanie mantís na sčítačke o jeden bit väčšej než je veľkosť mantisy (veľkosťou mantisy sa tu myslí počet bitov potrebných na uloženie mantisy aj so skrytou 1, čiže  $|1.M| + 1$ ).
4. Ak je to potrebné, vykoná sa normalizácia. Mantisa sa posunie o potrebný počet bitov doprava, resp. doľava, tak, aby bola v tvare  $1.M$ . Exponent sa zvýši, resp. zníži o daný počet bitov.
5. Na koniec výpočtu sa skontroluje hodnota exponentu. Ak je hodnota maximálna, došlo k pretečeniu výsledku. Ten sa nastaví podľa znamienka na kladné alebo záporné nekonečno. V opačnom prípade, ak je hodnota exponentu minimálna (nulová), došlo k podtečeniu, a výsledok je nastavený podľa znamienka na kladnú alebo zápornú nulu.

Hodnota operandu 1	Hodnota operandu 2	Výsledok sčítania
FP číslo	$+/- \infty$	$+/- \infty$
$+/- \infty$	$+/- \infty$	$+/- \infty$
$+\infty$	$-\infty$	NaN
NaN	ľubovoľná hodnota	NaN

Tabuľka 4.2: Výsledok operácie sčítania so špeciálnymi hodnotami.

### 4.2.2 Násobenie a delenie

Násobenie a delenie v pohyblivej rádovej čiarky sa počíta nasledovne:

$$X \times Y = (M_X \cdot M_Y) \cdot 2^{E_X + E_Y} \quad (4.5)$$

$$X \div Y = (M_X \div M_Y) \cdot 2^{E_X - E_Y}. \quad (4.6)$$

Postup výpočtu operácií násobenia a delenia podľa štandardu IEEE 754 ([12], [15]) je nasledovný:

1. Rovnako ako pri sčítaní, aj teraz sa na začiatku výpočtu skontroluje výskyt špeciálnych hodnôt oboch čísel podľa tabuľky 4.1. Ak ide o špeciálne číslo, výsledok sa určí podľa tabuľky 4.3 alebo 4.4. Pri operácii delenia je potrebné kontrolovať nepovolenú operáciu delenie nulou. Pokračuje sa nasledujúcim bodom.
2. Pri násobení sa hodnota exponentu vypočíta ako súčet exponentov, od ktorého sa odpočíta hodnota *BIAS*. Pri operácii delenia sa hodnota exponenta vypočíta ako rozdiel exponentov, ku ktorému je pripočítaná hodnota *BIAS*.
3. Výsledná mantisa sa rovná súčinu, resp. podielu mantís. Pri násobení je potrebné použiť násobičku, ktorej bitová šírka sa rovná dvojnásobku počtu bitov mantisy 1.*M*. Ak dôjde k pretečeniu alebo k podtečeniu mantisy, vykoná sa posun mantisy doprava, resp. doľava, a hodnota exponentu sa zvýši, resp. zníži.
4. Pokiaľ je to potrebné, prebehne normalizácia.
5. Na konci výpočtu sa skontroluje hodnota exponenta. Postupuje sa rovnako ako pri operácii súčtu: ak je hodnota exponenta maximálna, nastaví sa podľa znamienka na kladné alebo záporné nekonečno. V opačnom prípade, ak je hodnota exponenta minimálna (nulová), výsledok sa nastaví podľa znamienka na kladnú alebo zápornú nulu.

Hodnota operandu 1	Hodnota operandu 2	Výsledok násobenia
kladné/záporné FP číslo	+/- $\infty$	+/- $\infty$
nula	+/- $\infty$	NaN
+/- $\infty$	+/- $\infty$	+/- $\infty$
NaN	ľubovoľná hodnota	NaN

Tabuľka 4.3: Výsledok operácie násobenia so špeciálnymi hodnotami.

Hodnota operandu 1	Hodnota operandu 2	Výsledok násobenia
kladné/záporné FP číslo	+/- 0	+/- $\infty$
0	0	NaN

Tabuľka 4.4: Výsledok operácie delenia so špeciálnymi hodnotami.

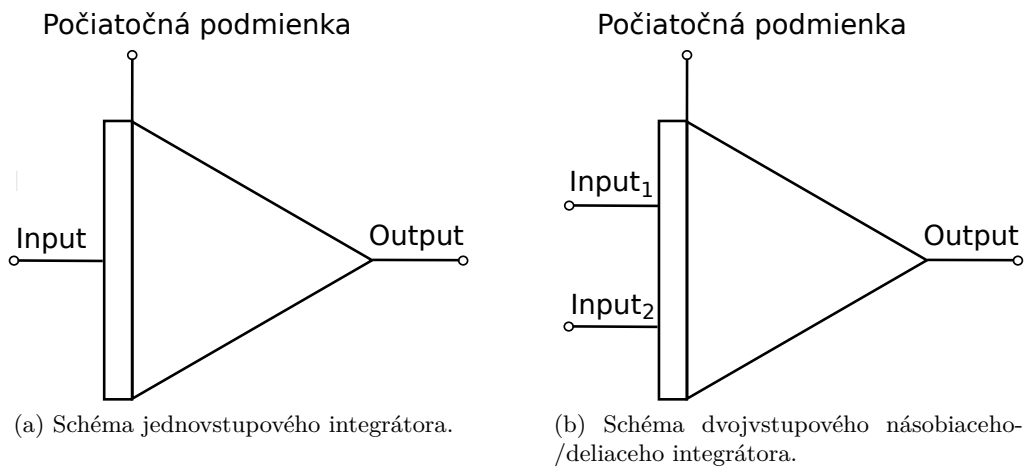
## Kapitola 5

# Numerické integrátory

Numerický integrátor je hardvérový komponent, ktorý slúži na výpočet numerickej integrácie. Podľa spôsobu výpočtu a komunikácie medzi komponentmi integrátora sa numerické integrátory delia na

- sériovo-sériové (sériové) integrátory (sériová komunikácia aj výpočet)
- sériovo-paralelné integrátory (sériová komunikácia a paralelný výpočet)
- paralelno-paralelné (paralelné) integrátory (paralelná komunikácia aj výpočet)

V tejto práci sa budeme zaoberať paralelnými numerickými integrátormi pre jednoduchosť a rýchlosť ich výpočtu. Ďalej môžeme numerické integrátory rozdeliť na jednovstupové a dvojevstupové. Jednovstupový integrátor vykonáva integráciu vstupovej hodnoty a posiela ju na výstup. Schéma tohoto typu integrátora je znázornená na obrázku 5.1a, a ako vyplýva z názvu, obsahuje jeden vstup pre vstupnú hodnotu, jeden vstup pre počiatočnú podmienku a jeden výstup pre výsledok výpočtu. Dvojevstupové integrátory obsahujú jeden vstup pre počiatočnú podmienku, dva vstupy pre prívod operandov a jeden výstup pre výsledok výpočtu. Schéma integrátora je znázornená na obrázku 5.1b.



Obr. 5.1: Schéma jednovstupového a dvojevstupového integrátora.

Tieto integrátory rozdeľujeme podľa použitej operácie na násobiace a na deliace integrátory. Násobiaci integrátor vykonáva deriváciu násobenia dvoch vstupných hodnôt. Deliaci

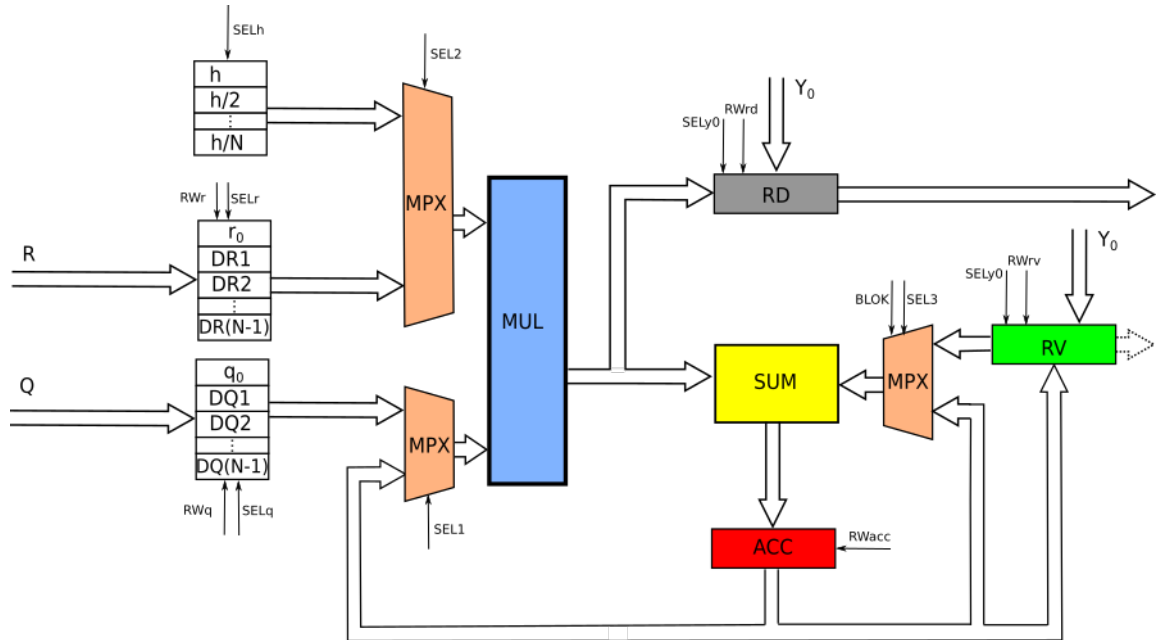
integrátor zasa vykonáva deriváciu delenia dvoch vstupných hodnôt. Výsledná hodnota je poslaná na výstup.

V nasledujúcich podkapitolách predstavíme jednotlivé návrhy paralelných násobiacich a paralelných deliacich integrátorov, oba typy v pevnej a v pohyblivej rádovej čiarke. Predstavíme taktiež jednovstupový paralelný integrátor v pevnej a v pohyblivej rádovej čiarke.

## 5.1 Návrh násobiaceho integrátora v pevnej rádovej čiarke

Násobiaci integrátor počíta rovnicu (3.6) pomocou (3.11) – (3.14). Na základe týchto rovníc bol vytvorený návrh paralelného násobiaceho integrátora, ktorý je na obrázku 5.2. Návrh vychádza z práce V. Závalu [14] a bol upravený a rozšírený o počet registrov  $DR(N-1)$  a  $DQ(N-1)$ , ktoré slúžia na ukladanie prichádzajúcich členov. Rozšírený bol aj počet registrov slúžiacich na uloženie kroku  $h$  integračnej metódy a jeho podielov. Počet týchto registrov je o jeden menší ako rád  $ORD$  použitej Taylorovej metódy ( $ORD = N$ ), keďže pri výpočte sa používajú predchádzajúce členy Taylorovej metódy. Rád metódy je vhodné a potrebné zvoliť vzhľadom na použitú aritmetiku a požadovanú presnosť.

Každý člen  $DY(N-1)$  obsahuje postupné delenie integračného kroku  $h$ . Tieto hodnoty sú predpočítané a uložené v sade registrov  $h$ . Pre optimalizáciu a pre ušetrenie miesta však nie je potrebné uložiť všetkých  $N$  hodnôt, stačí uložiť len tie hodnoty, ktorých deliteľom je nepárne číslo. Ostatné hodnoty je možné vypočítať jednoduchým posunom registra doprava, čo je vlastne delenie číslom 2. Je síce potrebné pridať riadiace signály na ovládanie posunu jednotlivých registrov, avšak výsledkom je zníženie počtu registrov  $h$  o polovicu. Počet operácií potrebných na výpočet sa nezvýši, keďže posun registrov je možné vykonávať v predstihu a paralelne s inými operáciami.



Obr. 5.2: Paralelno-paralelný násobiaci integrátor [14].

Skratka	Popis
<b>R, Q</b>	vstupné operandy integrátora
<b>Y0</b>	počiatočná podmienka
<b>N</b>	maximálny rád metódy
<b>h, h/2...h/N</b>	set registrov s integračným krokom metódy a s jeho podielmi
<b>r0, DR1, ...DR(N-1)</b>	set registrov jednotlivých členov vstupného operandu R
<b>q0, DQ1, ...DQ(N-1)</b>	set registrov jednotlivých členov vstupného operandu Q
<b>MUL</b>	paralelná násobička
<b>SUM</b>	paralelná sčítačka
<b>ACC</b>	akumulátor - register na priebežné ukladanie medzivýsledkov
<b>RD</b>	register na uloženie jednotlivých členov Taylorovej rady
<b>RV</b>	register na uloženie celkového výsledku
<b>MPX</b>	multiplexor 2-1
<b>SEL1, SEL2, SEL3</b>	riadiace signály multiplexorov
<b>SELh, SELr, SELq</b>	riadiace signály registrov
<b>SELy0</b>	riadiaci signál na zápis počiatočnej podmienky
<b>RWr, RWq</b>	povoľovacie signály setov registrov (READ/WRITE)
<b>RWrv, RWrd, RWacc</b>	povoľovacie signály registrov (READ/WRITE)
<b>BLOK</b>	blokový signál multiplexora - na výstupe 0

Tabuľka 5.1: Význam skratiek použitých v obrázku 5.2.

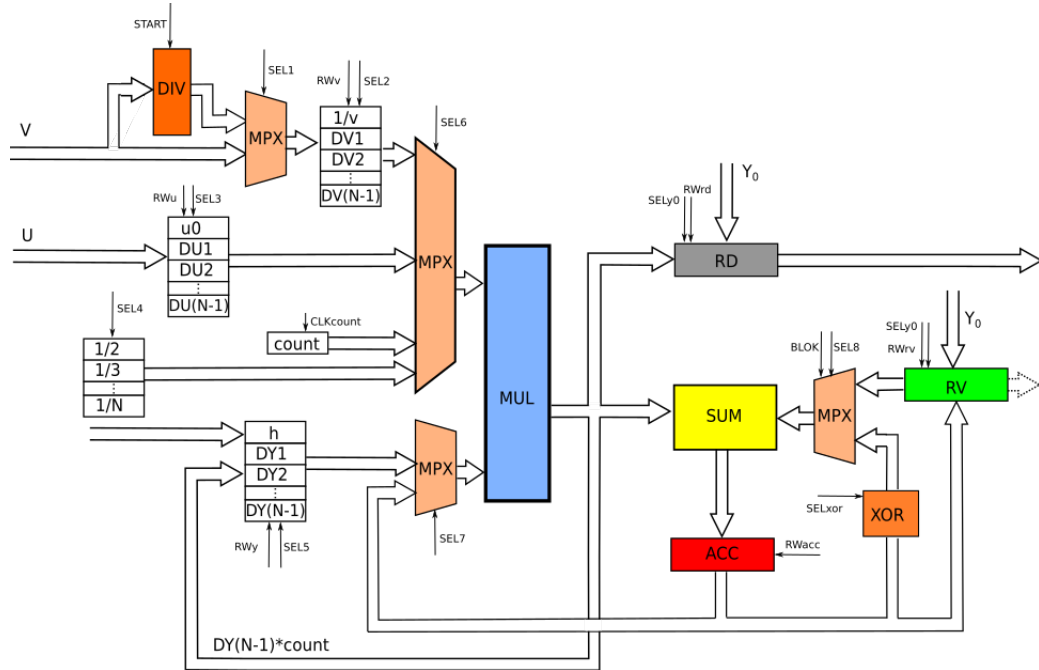
Na začiatku výpočtu sú pomocou signálu *RESET* vynulované všetky registre a nastaví sa potrebné signály na multiplexoroch *MPX*. Následne je do registrov *RV* a *RD* nahraná počiatočná podmienka, do registra *h* je nahraný integračný krok metódy a do registrov  $h/i$ ,  $i=2,3..N$  sú nahrané predpočítané hodnoty podielov. Povoľovacím signálom *EN* sa spustí výpočet. Ako prvé sa do registrov uložia vstupné hodnoty *R* a *Q*. Na začiatku každého cyklu výpočtu sú vstupné hodnoty ukladane postupne do ďalších registrov. Po uložení sa začne samotný výpočet. Hodnota aktuálne počítaného člena Taylorovej rady sa postupne ukladá do registra *RD* a odtiaľ je privedená na výstup integrátora. Do registra *RV* je ukladán súčet jednotlivých členov Taylorovej rady - čiže celkový výsledok derivácie vstupných hodnôt. Táto hodnota nás bude zaujímať napríklad vtedy, keď je integrátor "posledný" v zapojení, čiže jeho výstup nie je nikam pripojený, a teda nie je potrebné použiť hodnoty jednotlivých členov.

## 5.2 Návrh deliaceho integrátora v pevnej rádovej čiarke

Tento integrátor počíta rovnicu (3.16) pomocou členov (3.22) - (3.25). Podobne ako pri násobiacom integrátore bol z týchto rovníc vytvorený návrh paralelného deliaceho integrátora. Návrh vychádza z bakalárskej práce [4] a bol upravený podobne ako predošlý násobiaci integrátor pomocou zvýšenia počtu registrov  $DU(N-1)$  a  $DV(N-1)$ . Taktiež sa zvýšil počet registrov  $DY(N-1)$ , ktoré slúžia na uloženie jednotlivých členov Taylorovej rady, keďže je na výpočet nasledujúceho člena použitý predchádzajúci člen. Set týchto registrov okrem hodnôt  $DY(N-1)$  obsahuje register na uloženie kroku *h* integračnej metódy. Deliaci integrátor, na rozdiel od násobiaceho integrátora, nepotrebuje počas výpočtu hodnoty podielov kroku numerickej metódy, a teda stačí uložiť len jednu hodnotu *h*. Deliaci integrátor však používa set registrov, ktorý obsahuje hodnoty  $1/N$ . Podobne ako pri násobiacom



integrátore, aj v deliacom integrátore je možné zmenšiť počet týchto registrov na polovicu s využitím aritmetického posunu (operácia *shift*). Register *count* obsahuje *counter*, ktorý sa postupne inkrementuje v každom cykle výpočtu, a s ktorým sa násobí hodnota  $DY(N-1)$ . Vyplyva to z rovníc (3.22) - (3.25) a (3.26), kde sa pri výpočte každého člena Taylorovej rady hodnota  $DY(N-k)$  násobí hodnotou  $N-k$ . Výsledná vynásobená hodnota je uložená opätovne do daného registra  $DY(N-1)$  a znova použitá v ďalšom výpočte. Tým sa ušetrí operácia násobenia v ďalšom cykle výpočtu.



Obr. 5.3: Paralelno-paralelný deliaci integrátor [4].

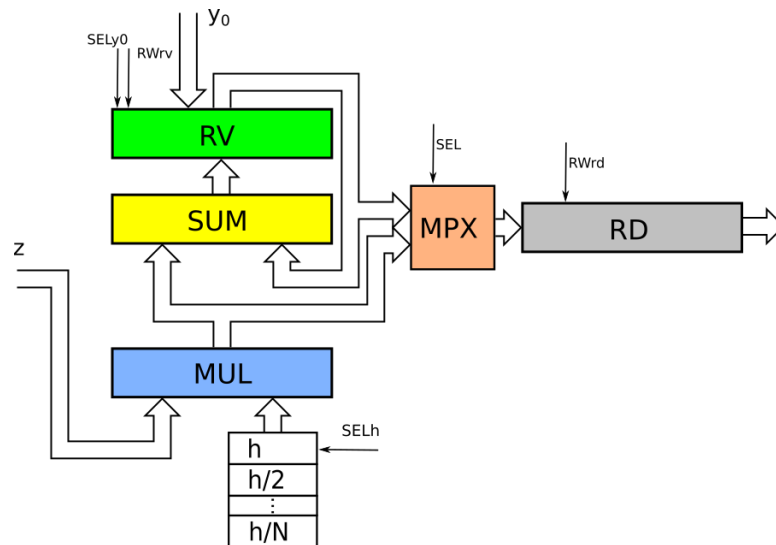
Skratka	Popis
V, U	vstupné operandy integrátora
$1/v, DV1, \dots, DV(N-1)$	set registrov jednotlivých členov vstupného operandu V
$u0, DU1, \dots, DU(N-1)$	set registrov jednotlivých členov vstupného operandu U
count	čítač cyklov výpočtu - N
$1/2, 1/3, \dots, 1/N$	set registrov s jednotlivými podielmi
$h, DY1, \dots, DY(N-1)$	set registrov s krokom metódy a s vypočítanými členmi
DIV	delička
XOR	invertovanie hodnoty v dvojkovom doplnku (xor +1)
MPX	multiplexor 2-1, 4-1
START	povoľovací signál na spustenie výpočtu
SEL1, SEL6, SEL7, SEL8	riadiace signály multiplexorov
SEL2, SEL3, SEL4, SEL5	riadiace signály registrov
SELxor	signál na invertovanie vstupnej hodnoty
CLKcount	signál na zvýšenie hodnoty čítača
RWv, RWu, RWy	povoľovacie signály setov registrov (READ/WRITE)

Tabuľka 5.2: Význam skratiek v obrázku 5.3.

Podobne ako pri násobiacom integrátore, aj pri paralelnom deliacom integrátore sú na začiatku výpočtu vynulované registre pomocou signálu *RESET* a nastavené signály multiplexorov *MPX*. Ďalej je do registrov *RD* a *RV* nahraná počiatočná podmienka a do registra *h* je nahraný integračný krok. Do registrov  $1/n$  sú uložené predvypočítané konštanty. Po prijatí hodnôt *U* a *V* sa začne výpočet. Hodnota *V* je privedená do deličky *DIV* a spustí sa výpočet  $1/v$  s použitím deliaceho algoritmu *SRT*. Delenie je realizované len raz počas celého výpočtu, keďže ide o veľmi náročnú operáciu. Po skončení operácie delenia je výsledok uložený do setu registrov *DVN*, a to konkrétne do registra  $1/v$ . V ďalšom cykle výpočtu je vstupná hodnota *V* privedená priamo do setu registrov *DVN* bez delenia. Hodnota *U* je uložená do setu registrov *DUN*. Pre optimalizáciu je paralelne s delením realizovaný výpočet násobenia  $uh$ . Po skončení výpočtu je výsledok aktuálneho člena uložený do registra *RD* a privedený na výstup deliaceho integrátora. Suma jednotlivých členov sa postupne ukladá v každom cykle do registra *RV*, z ktorého je možné získať hodnotu privedenú na výstup, rovnako ako aj v násobiacom integrátore.

### 5.3 Návrh jednovstupového integrátora

Tento integrátor bol navrhnutý podľa rovníc (3.5), na základe ktorých integrátor počíta diferenciálnu rovnicu (3.1). Z toho vyplýva, že vykonáva numerickú integráciu vstupnej hodnoty, a následne výslednú hodnotu posiela na výstup. Schéma integrátora je na obrázku 5.4 a vychádza z práce J. Opálku [6]. Na naštartovanie výpočtu slúži počiatočná podmienka. Tá je nahraná na začiatku výpočtu do registra *RV*. Následne je cez multiplexor *MPX* privedená a uložená do registra *RD*. V sete registrov *h* je uložený integračný krok metódy a jeho podiely. Rovnako ako aj v predchádzajúcich prípadoch, aj tu je možné počet registrov znížiť na polovicu. Vstupnú hodnotu *Z* nie je potrebné ukladať do registrov, keďže sa nepoužíva pri výpočte ďalších členov. Jednotlivé hodnoty členov sú ukladané do registra *RV* a poslané na výstup integrátora. Ak chceme získať výslednú hodnotu Taylorovej rady rovnice (3.1), stačí priviesť hodnotu z registra *RV* cez multiplexor *MPX* do registra *RD*, kde je hodnota sprístupnená na výstupe integrátora.

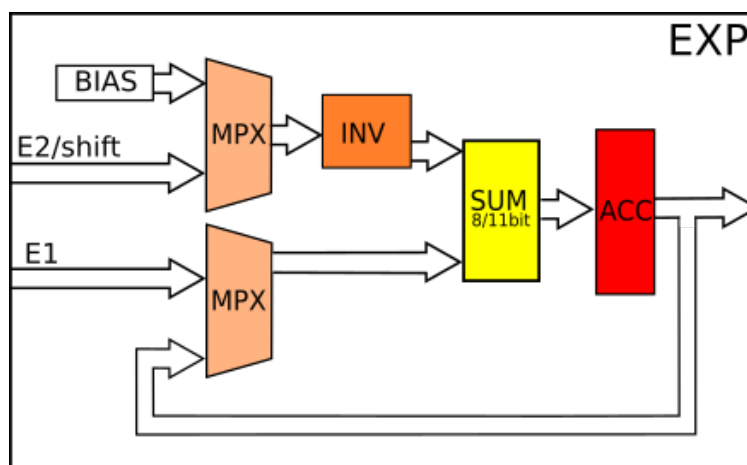


Obr. 5.4: Paralelný jednovstupový integrátor v pohyblivej rádovej čiarke.

## 5.4 Návrh komponentu na spracovanie exponentov

V pohyblivej rádovej čiarke sú operácie násobenia, delenia a sčítania (odčítania) zložitejšie, nakoľko je k týmto operáciám pridaná aj práca s výpočtom znamienka, exponentu a mantisy. Výpočet exponentov je možné vykonávať v každom z komponentov (násobička, delička, sčítačka) samostatne, alebo pomocou jedného komponentu spoločného pre tieto komponenty. Pri spoločnom komponente je možné znížiť priestorovú zložitosť zapojenia. Tým sa však zvýši náročnosť synchronizácie a riadenia kontrolóra. V deliacom integrátore to môže spôsobiť zdržanie výpočtu vtedy, keď sa operácie delenia a násobenia vykonávajú paralelne. Avšak táto situácia nastane len raz, na začiatku výpočtu. Predošlé popísané návrhy násobiaceho, deliaceho a jednovstupového integrátora je možné použiť ako pri implementácii v pevnej rádovej čiarke, tak aj v pohyblivej rádovej čiarke. Integrátory v pohyblivej rádovej čiarke vychádzajúce z popísaných návrhov pracujú s exponentami v každej operácii samostatne. V nasledujúcich sekciách si popíšeme druhý variant spracovania exponentu so spoločným komponentom.

Blokové schémy zapojenia integrátorov pracujúcich v pohyblivej rádovej čiarke vychádzajú z predchádzajúcich návrhov. Boli však rozšírené o prácu so znamienkami, s mantisou a s exponentmi. Samotný výpočet exponenta sa deje v spoločnom komponente *EXP*, ktorého návrh je zobrazený na obrázku 5.5.



Obr. 5.5: EXP - blok pracujúci s exponentmi.

Skratka	Popis
E1	vstupná hodnota väčšieho exponenta
E2/shift	vstupná hodnota menšieho exponenta/hodnota posunu
BIAS	hodnota 127 alebo 2047
SUM (8/11bit)	paralelná 8 alebo 11 bitová sčítačka
INV	invertor

Tabuľka 5.3: Význam skratiek použitých v obrázku 5.5.

Komponent *EXP* slúži na vykonávanie výpočtov s exponentmi, ktoré sú popísané v sekciách 4.2.1 a 4.2.2. Obsahuje dva vstupy, do ktorých sú privedené jednotlivé exponenty z komponentov *DIV*, *MUL* a *SUM*, a jeden výstup. Celý komponent *EXP* pracuje na počte bitov, ktorý sa rovná veľkosti exponenta, ktorá je závislá od použitej aritmetiky. Pri jednoduchšej presnosti (32 bitov) je veľkosť exponenta 8 bitov a pri dvojitej presnosti (64 bitov) je veľkosť exponenta 11 bitov. Komponent teda obsahuje buď 8-bitovú alebo 11-bitovú sčítačku a ostatné komponenty v rovnakej bitovej šírke. Pri vykonávaní operácie súčiny alebo rozdielu slúži komponent *EXP* na výpočet rozdielu exponentov. Výsledná hodnota rozdielu je privedená naspäť do sčítačky *SUM* cez register *ACC*. O túto hodnotu je následne v sčítačke posunutá hodnota mantisy menšieho čísla doprava. Výsledná hodnota mantisy je potom súčet/rozdiel mantisy väčšieho čísla a posunutej mantisy. Znamienko a exponent sa rovnajú hodnote väčšieho čísla z počítaných operandov. Pri operácii delenia alebo násobenia slúži komponent *EXP* na výpočet výsledného exponenta. Podľa typu operácie sa vykoná súčet alebo rozdiel prijatých exponentov. Táto hodnota je uložená v registri *ACC* a privedená naspäť do sčítačky  $SUM_{8/11bit}$  s hodnotou z registra *BIAS*. Vykoná sa súčet alebo rozdiel týchto hodnôt. Získame tak hodnotu nového exponenta, ktorá je privedená naspäť do násobičky *MUL* alebo deličky *DIV*. Ak je však po delení alebo násobení potrebné vykonať posun mantisy, hodnota vypočítaného exponenta je privedená z *ACC* cez multiplexor *MPX* naspäť do sčítačky  $SUM_{8/11bit}$ , a na vstup *E2/Shift* je privedená hodnota posunu. Vykoná sa rozdiel hodnôt, čím vykonáme posun exponenta. Následne je výsledná hodnota privedená do násobičky *MUL* alebo deličky *DIV* a je použitá ako nová hodnota exponentu.

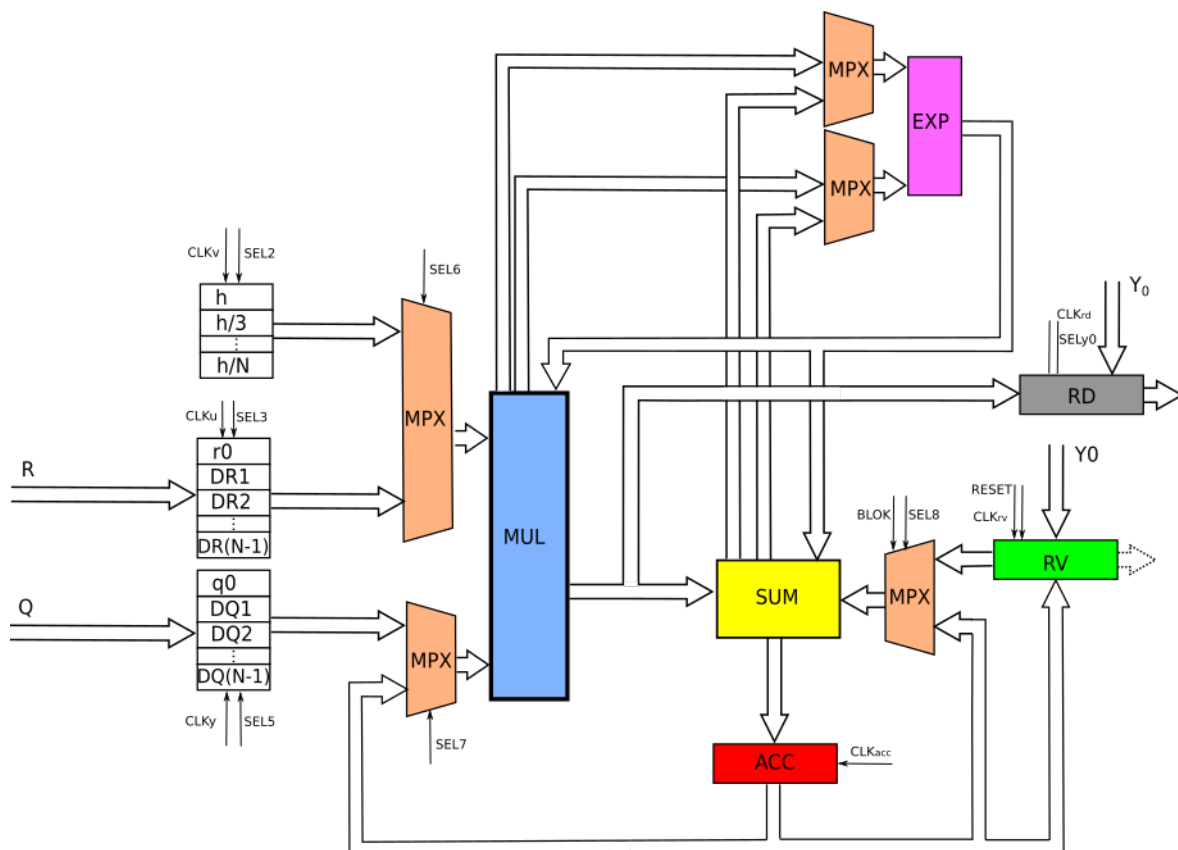
Schéma pre prehľadnosť neobsahuje znázornenie zapojenia výpočtu znamienka a rozdelenie operandov na znamienko, exponent a na mantisu, a ich opätovné zloženie. Tieto operácie sa vykonávajú v komponentoch *DIV*, *MUL* a *SUM*.

## 5.5 Návrh násobiacieho integrátora v pohyblivej rádovej čiarke

Návrh paralelného násobiaceho integrátora v pohyblivej rádovej čiarke so spoločným komponentom *EXP* je na obrázku 5.2. Čísla v pohyblivej rádovej čiarke môžeme zobrazit presnejšie ako čísla v pevnej rádovej čiarke, čiže pri uložení malých čísel v pohyblivej rádovej čiarke dochádza k menšej zaokrúhľovacej chybe. Z tohoto dôvodu je možné počítať Taylorovu radu s použitím väčšieho počtu členov a zvýšiť tak presnosť výpočtu. Integrátory v pohyblivej rádovej čiarke teda môžu počítať viac členov Taylorovej rady, čiže sa počet registrov  $DR(N-1)$  a  $DQ(N-1)$  zvýši. Set registrov  $h/i, i = 2, 3..N$  obsahuje  $N-1$  registrov. Počet týchto registrov je možné znížiť rovnako ako pri integrátoroch v pevnej rádovej čiarke.

Spôsob výpočtu paralelného násobiaceho integrátora v pohyblivej rádovej čiarke so spoločným komponentom *EXP* je takmer rovnaký ako pri násobiacom integrátore v pevnej rádovej čiarke. Rozdielne sa vykonávajú operácie násobenia a sčítania. Po privedení hodnôt na vstupy násobičky *MUL* alebo sčítačky *SUM* sú čísla v FP uložené do pomocných registrov v týchto komponentoch. Z týchto registrov sú jednotlivé časti FP čísla roz distribuované do samostatných výpočtových obvodov. V komponente *MUL* sú znamienka privedené ku komponentu *XOR*, ktorý vykonáva nonekvivalenciu. V komponente *SUM* sa výsledné znamienko rovná znamienku väčšieho čísla zo vstupných hodnôt. Exponenty, ako je popísané

vyššie, spracúva komponent *EXP*. Mantisy sú privedené do paralelnej násobičky alebo sčítačky. Veľkosť výsledku násobičky je dvojnásobná veľkosti mantisy so skrytou jednotkou, t. j. 48 bitov pri jednoduchovej presnosti a 106 bitov pri dvojitej presnosti. Veľkosť sčítačky sa rovná veľkosti mantisy so skrytou jednotkou, t. j. 24 alebo 53 bitov. Výsledné hodnoty jednotlivých častí čísla v pohyblivej rádovej čiarkke sú na výstupe spojené do jedného čísla a sú uložené do výstupného registra daného komponentu (*MUL* alebo *SUM*). Následne sú poskytnuté na jeho výstupe k ďalšiemu výpočtu.

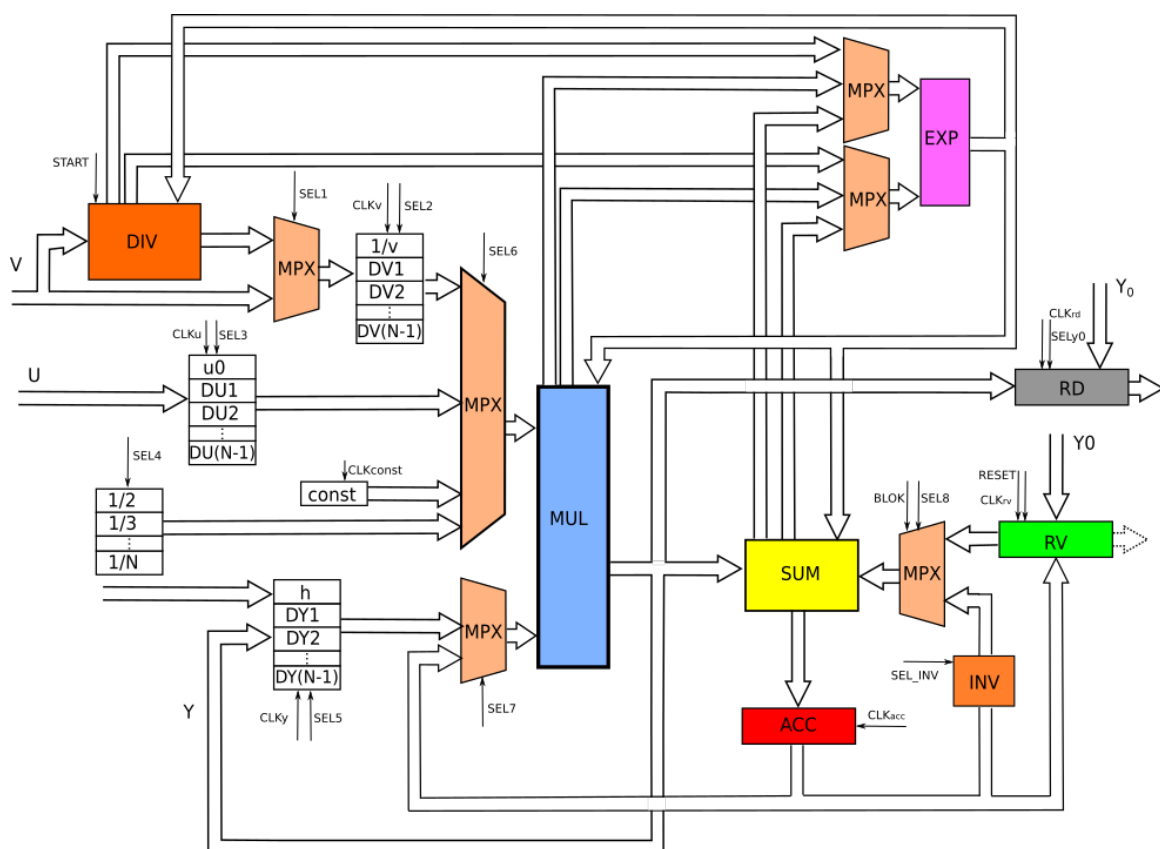


Obr. 5.6: Paralelno-paralelný násobiaci integrátor v pohyblivej rádovej čiarkke.

## 5.6 Návrh deliaceho integrátora v pohyblivej rádovej čiarkke

Paralelný deliaci integrátor v pohyblivej rádovej čiarkke so spoločným komponentom *EXP* je najzložitejší z navrhnutých integrátorov. Obsahuje všetky spomínané operácie: sčítanie, odčítanie, násobenie a delenie; a všetky vykonáva v pohyblivej rádovej čiarkke, v ktorej je práca s exponentami vykonávaná v jednom komponente *EXP*. Operácie násobenia a sčítania sa vykonávajú rovnako ako v paralelnom násobiacom integrátore v pohyblivej rádovej čiarkke. Operácia delenia sa vykonáva iba raz za celý výpočet, a to paralelne s operáciou násobenia, ako aj v deliacom integrátore s FX aritmetikou. Tu však môže dôjsť ku kolízii v použití komponentu *EXP*, keďže delenie je náročnou operáciou, pričom dĺžka výpočtu závisí na použití daného algoritmu. Podľa zvoleného algoritmu je potom potrebné určiť, ktorá

operácia bude mať prednosť v použití komponentu *EXP*. Po uvoľnení *EXP* násobičkou *MUL* je komponent *EXP* pridelený deličke *DIV* na výpočet exponentov. Ak je použitý zdĺhavejší algoritmus pri výpočte delenia (napr. SRT algoritmus použitý v deliacom integrátoře vo FX aritmetike), čiže ak operácia delenia trvá dlhší čas ako operácia násobenia, je vhodnejšie prenechať komponentu *EXP* najprv operáciu násobenia. Ak je deliaci algoritmus veľmi rýchly a čas výpočtu je podobný času výpočtu operácie násobenia, tak potom nezáleží, ktorý z komponentov bude ako prvý využívať *EXP*. Po skončení delenia je podiel uložený do registra  $1/v$  a pokračuje sa v ďalšom výpočte. Výpočet prebieha podobne ako pri deliacom integrátoře bez komponentu *EXP*. Rozdielne sa vykonávajú operácie delenia, násobenia a sčítania. Ďalší rozdiel je v komponente *XOR*, ktorý bol nahradený komponentom *INV*. Prevrátenie hodnoty čísla sa v pevnej rádovej čiarke použitím doplnkového kódu deje operáciou *XOR* a pričítaním jednotky, ako to je v predchádzajúcom deliacom integrátoře. V pohyblivej rádovej čiarke sa však invertovanie hodnoty čísla vykonáva jednoduchým invertovaním znamienkového bitu *S* vstupného operandu. Toto invertovanie sa vykonáva v spomínanom komponente *INV*.



Obr. 5.7: Paralelno-paralelný deliaci integrátor v pohyblivej rádovej čiarke.

## 5.7 Sústava diferenciálnych rovníc

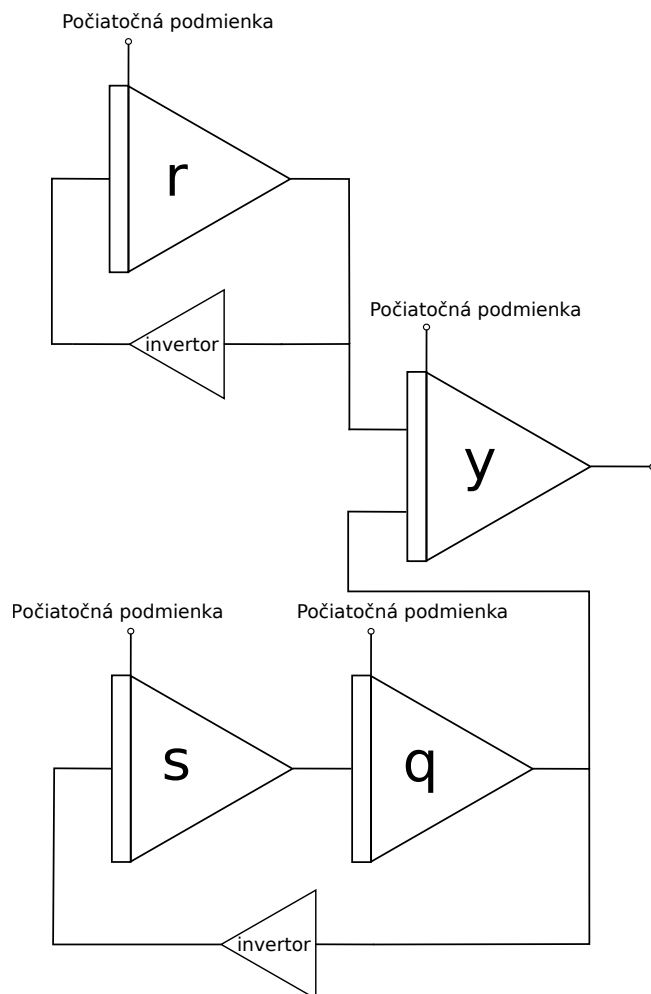
Integrátory, ktoré sme si popísali, je možné medzi sebou zapojiť, a riešiť tak diferenciálnu rovnicu. Po zapojení integrátorov nebude nutné predvypočítať vstupné hodnoty, stačí len zadať počiatočné podmienky na naštartovanie výpočtu, a všetky ostatné hodnoty sa dopočítajú. Zvoľme konkrétnu diferenciálnu rovnicu s operáciou násobenia v tvare

$$y' = \sin(t)e^{-t}, \quad y(0) = 1. \quad (5.1)$$

Rovnicu prevedieme do sústavy diferenciálnych rovníc:

$$\begin{aligned} y' &= qr, & y(0) &= 1 \\ r' &= -r, & r(0) &= 1 \\ q' &= s, & q(0) &= 0 \\ s' &= -q, & s(0) &= 1. \end{aligned} \quad (5.2)$$

Zo sústavy rovníc vytvoríme schému zapojenia integrátorov, znázornenú na obrázku 5.8.



Obr. 5.8: Schéma zapojenia integrátorov [4].

Schému tvoria tri jednovstupové integrátory  $r$ ,  $q$  a  $s$ , dvojevstupový násobiaci integrátor  $y$  a dva invertory. Invertor je komponent, ktorý invertuje prichádzajúcu hodnotu. Integrátor  $r$  a invertor tvoria spolu funkciu  $e^{-t}$ . Integrátory  $r$ ,  $s$  a invertor tvoria spolu funkciu  $\sin(t)$ . Nakoniec sú tieto hodnoty privádzané na vstup násobiaceho integrátora  $y$ , ktorý počíta diferenciálnu rovnicu 5.1. Zo schémy zapojenia je možné následne vytvoriť VHDL kód a vykonať výpočet na FPGA. Takýto postup je vhodný pre rozsiahle diferenciálne rovnice s väčším počtom operácií, ktorých výpočet by bol vykonávaný veľmi často. Ako vhodnejší spôsob použitia sa ponúka prepojovacia sieť medzi rôznymi integrátormi. Pred výpočtom by sa sieť nakonfigurovala podľa danej diferenciálnej rovnice a výpočet by mohol prebiehať bez nutnosti syntézy. Tento variant je síce priestorovo náročný, ale ponúka variabilitu výpočtu. V ďalšej kapitole si predstavíme implementáciu popísaných integrátorov a taktiež implementáciu sústavy diferenciálnych rovníc pomocou metódy bez použitia prepojovacej siete.



## Kapitola 6

# Popísanie integrátorov vo VHDL

Jednotlivé integrátory predstavené v kapitole 5 boli popísané v jazyku VHDL v prostredí Xilinx ISE a následne odsimulované v nástroji Model Sim s pomocou použitia *testbenchu*. Potom boli niektoré z integrátorov otestované na programovateľnom hradlovom poli FPGA. Pre priestorovú zložitosť paralelných integrátorov bola snaha implementovať integrátory na rozsiahlejšom čipe FPGA, a to konkrétne na VIRTEX-5 s čipom XC5VSX50T od spoločnosti Xilinx Inc. Tento prípravok môžeme vidieť na obrázku 6.5a v závere tejto kapitoly. Pre prácu s týmto čipom je potrebná plná verzia programu Xilinx ISE s licenciou. Poskytnutie licencie od VUT FIT pre študenta však nie je možné. Riešením bolo použitie licencie pomocou pripojenia sa na fakultnú sieť cez VPN. Následne bol integrátor implementovaný na VIRTEX-5, avšak skončenie výpočtu bolo možné zistiť len rozsvietením diódy, čo je nepoužiteľný spôsob na získanie hodnoty výsledku. Preto bolo snahou výsledok zobrazit' na LCD displeji, prípadne vytvorit' komunikáciu cez terminál. Spreádzkovanie displeja bolo podľa rôznych návodov neúspešné a pre komunikáciu s terminálom bolo potrebné doinštalovať ďalšie programy. Vzhľadom na dané komplikácie bolo teda nutné použiť iný čip, a to konkrétne FPGA Spartan 3 XC3S50, ktorý je použitý v študijnej pomôcke FitKit 2.0. Veľkosť tohto FPGA je však veľmi malá, a ani jeden z navrhnutých integrátorov nie je možné na tento čip naprogramovať. Po konzultácii s Ing. Václavom Šimekom sme začali pracovať s prípravkom FitKit 2.0 s vymenením FPGA za väčší, a to za Spartan 3 XC3S400. FitKit 2.0 je znázornený na obrázku 6.5b na konci tejto kapitoly. Práca s týmto FPGA je kompatibilná s predchádzajúcim čipom, a preto bolo možné na programovanie použiť nástroj QDevKit. Ide o multiplatformový terminálový program pre jednoduchú prácu s FitKitom. Všetky parametre sú už v programe nastavené a komunikácia prebieha cez USB namiesto emulovaného sériového portu, ako je to pri iných terminálových programoch. Informácie o FitKite, o programe QDevKit a taktiež návody na vytvorenie vlastnej aplikácie boli čerpané z webstránky FitKitu [13].

Na FitKite bol odskúšaný jednovstupový integrátor s 32-bitovou fixed point a floating point aritmetikou. Tento integrátor bol vybraný pre najmenšiu priestorovú zložitosť. Ďalej bol implementovaný integrátor počítajúci rovnicu 6.1. Integrátor pracuje na 32 bitoch a používa FX aritmetiku. Rovnica je počítaná Taylorovou radou 4. rádu. Ostatné integrátory a verzie integrátorov nie je možné nahráť na FPGA, ktoré je použité vo FitKite.

Jednotlivé časti integrátorov boli popísané samostatne a následne bola vytvorená štruktúra vzájomného prepojenia, ktorá tvorí celý integrátor. Pri každom integrátore sú jednotlivé kroky výpočtu riadené kontrolórom. Ten je tvorený cyklickým konečným automatom. Cyklus vyplýva z rovníc (3.15) a (3.26), keďže výpočet jednotlivých členov je podobný, hoci sa počet perácií zväčšuje. Samozrejme, po skončení výpočtu je možné pomocou signálu

*RESET* prejsť do počiatočného stavu, a zadaním novej počiatočnej podmienky začať nový výpočet. Ďalej podrobnejšie popíšeme jednotlivé integrátory.

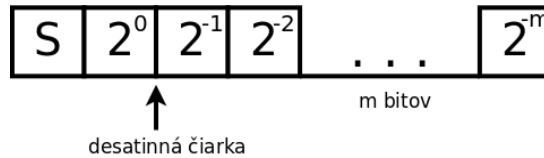
## 6.1 Integrátory v pevnej rádovej čiarke

Všetky integrátory v pevnej rádovej čiarke (t. j. deliaci, násobiaci a jednovstupový integrátor) kódujú čísla v dvojkovom kóde. Použité sčítačky a násobičky sú paralelné a výpočet operácie trvá jeden hodinový takt. Pri výpočte je však potrebné použiť dva takty, aby došlo k ustáleniu signálov: jeden na nastavenie signálov a druhý na uloženie výsledku.

### Násobiaci integrátor

Paralelný dvojevstupový násobiaci integrátor vo fixpoint aritmetike bol implementovaný v dvoch verziách, a to na 32 a 64 bitoch. Veľkosť použitej aritmetiky je však možné jednoducho zväčšiť, keďže celý kód je písaný genericky. Taktiež je možné zmeniť počet cyklov výpočtu, čiže počet počítaných členov Taylorovej rady, avšak v závislosti od počtu cyklov  $N$  je nutné upraviť počet registrov v registrových setoch. Pri násobiacom integrátore sú to sety registrov  $DQ(N-1)$ ,  $DR(N-1)$  a počet registrov  $h/i$ , kde  $i = 2..N$ . Pri deliacom integrátore je potrebné zväčšiť počet registrov  $DV(N-1)$ ,  $DU(N-1)$  a  $1/i$ , kde  $i = 2..N$ . S tým súvisí potreba následného zväčšenia bitovej šírky riadiacich signálov označovaných s prefixom *SEL*. Bitovú šírku signálov je vhodné zvoliť z počtu  $N$  registrov podľa vzťahu  $\text{šírka registrov} = \log_2 N$  a využiť tak celú šírku a kombinácie binárneho kódovania.

Pre 32-bitovú i pre 64-bitovú verziu bola použitá FX aritmetika, ktorá je názorne zobrazená na obrázku 6.1. Aritmetika je tvorená jedným znamienkovým bitom, 1 bitom pred desatinnou čiarkou a s  $m$  bitmi za desatinnou čiarkou. Pri jednoduchej presnosti sa  $m$  rovná 30 bitom a pri dvojitej presnosti sa  $m$  rovná 62 bitom. Výpočet počíta s Taylorovou metódou 8. rádu, a teda výpočet prebieha až po 8. člen Taylorovej rady, t. j. DY8.



Obr. 6.1: FX aritmetika numerického integrátora.

Násobiaci integrátor je ovládaný pomocou kontrolóra. Jednotlivé stavy kontrolóra a nastavenia signálov sú uvedené v tabuľke 6.1. Kontrolór obsahuje 9 stavov a dva čítače *count* a *countQ*. Čítač *count* udáva číslo aktuálneho cyklu a *countQ* je v každom cykle nastavený na hodnotu *count* a postupne sa dekrementuje. Výpočet začína nastavením signálu *reset* na logickú jednotku. To spôsobí prepnutie násobiaceho integrátora do stavu s rovnakým názvom, čiže *Reset*. V ňom sa integrátor uvedie do počiatočného stavu. Vynulujú sa počítadlá a čaká sa na povoloovací signál *EN*, ktorým sa prepne integrátor do stavu *Init*. V tomto stave je nahraná počiatočná podmienka a sú uložené vstupné hodnoty. Výpočet prejde do stavu *MulDQxDRx*. V ňom sú vynásobené jednotlivé členy  $DQi$  a  $DRj$ , kde  $i, j$  značí aktuálne počítaný člen. Z rovnice (3.15) vyplýva, že sa vykonáva násobenie kombinácií týchto členov. Na to slúžia spomenuté čítače *count* a *countQ*. Na základe čítačov sa nastavujú signály

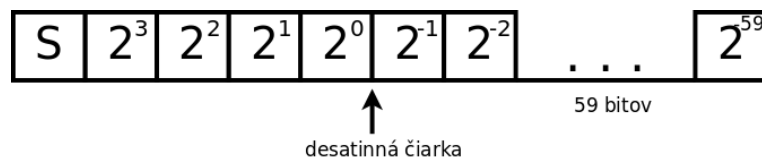
*SELR* a *SELQ*. Po vynásobení členov je výsledok v stave *WriteAcc* uložený do akumulátora. Zo stavu *WriteAcc* je integrátor prepnutý buď do stavu *SetupCounter*, alebo do stavu *MulAccH\_RV*. V stave *SetupCounter* sa dekrementuje počítadlo *countQ* a začína sa násobenie ďalšej kombinácie členov *DQi* a *DQj*. Ak sú vynásobené všetky kombinácie týchto členov, stačí potom už len vynásobiť obsah akumulátora hodnotou  $h/N$ . Integrátor je teda prepnutý do stavu *MulAccH\_RV* a vykoná sa operácia násobenia. Výsledok je uložený do registra *RD* (nejde o variantu „posledného“ integrátora v zapojení). Potom je v stave *SaveToAcc* sčítaná hodnota s hodnotou registra *RV* a následne je uložená do akumulátora. V ďalšom stave *SaveToRv* je výsledok v *ACC* uložený do registra *RV*. Výpočet končí v stave *Result*, v ktorom je nastavený signál *OEN* (output enable) na logickú jednotku. Po nastavení povolovacieho signálu *EN* na logickú jednotku je výpočet znova spustený s novými hodnotami.

### Deliaci integrátor

Paralelný deliaci integrátor vo fixpoint aritmetike bol implementovaný, podobne ako násobiaci integrátor, v 32-bitovej a v 64-bitovej verzii. Pri návrhu integrátorov sa kládol dôraz nielen na rýchlosť výpočtu, ale aj na priestorovú zložitosť. Z rovnice (3.26) vyplýva, že členy *DYN* sú násobené číslom  $N - k$ , kde  $k$  je minimálne 1, a teda najväčšie číslo je  $N - 1$ . Aby bolo možné toto číslo zobraziť, bolo potrebné zvoliť vhodnú aritmetiku, t. j. vhodný počet čísel pred a za desatinnou čiarkou. Čím väčší počet bitov sa nachádza pred desatinnou čiarkou, tým viac sa zväčšuje interval zobraziteľných čísel, avšak znižuje sa počet bitov za desatinnou čiarkou. Pri znižovaní počtu čísel za desatinnou čiarkou sa znižuje hustota zobrazenia čísel a v konečnom dôsledku aj ich presnosť. V dvojkovom doplnkovom kóde pri 1 bite pred desatinnou čiarkou sme schopní vypočítať deliacim integrátorom maximálne dva členy Taylorovej rady. Pri dvoch bitoch sú to 4 členy, pri troch bitoch 8 členov Taylorovej rady a pri štyroch bitoch 16 členov Taylorovej rady. To je možné vyjadriť ako  $N = 2^w$ , kde  $N$  - počet členov,  $w$  - bitová šírka. Keďže pri použití Taylorovej metódy 4. rádu nemusí byť riešenie dostatočne presné, 32-bitová verzia integrátora počíta s Taylorovou metódou 8. rádu, a teda je použitá aritmetika s jedným znamienkovým bitom, s tromi bitmi pred desatinnou čiarkou. Ostatných 28 bitov tvorí časť za desatinnou čiarkou. Vyšší rád metódy nie je použiteľný, keďže hodnoty vyšších členov by boli veľmi malé, a teda nezobraziteľné v danej aritmetike. Táto aritmetika je zobrazená na obrázku 6.2.



Obr. 6.2: FX aritmetika deliaceho integrátora na 32 bitoch.



Obr. 6.3: FX aritmetika deliaceho integrátora na 64 bitoch.

V 64-bitovej verzii deliaceho integrátora sa ponúka väčšia bitová šírka a teda aj presnosť, preto bola veľkosť použitých bitov pred desatinnou čiarkou zväčšená o jeden bit. Za desatinnou čiarkou sa potom nachádza 59 bitov. V tejto verzii deliaceho integrátora sme teda schopní vypočítať v danej aritmetike až 16 členov Taylorovej rady. Použitá aritmetika je na obrázku 6.3.

Postup výpočtu deliaceho integrátora je podobný ako v násobiacom integrátore. Jednotlivé stavy kontrolóra a nastavenie signálov sú uvedené v tabuľke 6.3. Začiatok výpočtu je rovnaký ako pri integrátore s operáciou násobenia. Integrátor je v počiatočnom stave, t. j. v stave *Reset*. Z neho prejde do stavu *Init*. Následne je v stave *MulUH* násobená hodnota člena  $DU_i$  s integračným krokom  $h$ . Výsledok je uložený do akumulátora *ACC*. Výsledok násobenia je možné uložiť už v tomto kroku, keďže nastavenie signálov multiplexorov a výpočet násobenia bolo možné začať už v stave *Init*. Paralelne s operáciou násobenia je signálom *START* zahájené delenie  $1/v$ , kde  $v$  je vstupná hodnota operandu. Operácia delenia sa vykonáva pomocou SRT algoritmu s radixom 2. Bližšie informácie o danom algoritme a o implementácii môžeme nájsť v [4]. V stave *WaitDiv* je integrátor počas niekoľkých cyklov, až kým sa neskončí operácia delenia. Delička po vykonaní operácie delenia nastaví výstupný signál *done* na 1, čím umožní prechod integrátora do nasledujúceho stavu *SaveDiv*. V ňom je hodnota delenia uložená do registra  $1/v$ . Tým dostaneme hodnotu prvého člena Taylorovej rady. Hodnota je sčítaná s hodnotou registra *RV* a uložená späť do registra.

Stav	SELy0	SEL1	SEL2	SEL3	SELQ	SELR	SELH	RWAcc	RWrv	BLOK	
Reset	0	0	0	0	0	0	0	0	0	0	count = 0; countQ = 0;
Init	1	X	X	X	X	X	X	0	1	X	count = 0; countQ = 0;
MulDQxDRx	X	1	0	0	countQ	count-countQ	X	0	0	countQ=count ? 1 : 0	count < N ? WriteToAcc : Result
WriteToAcc	X	1	0	0	countQ	count-countQ	X	1	0	countQ=count ? 1 : 0	countQ = 0 ? MulAccH_RV : SetupCounter
SetupCounter	X	X	X	X	X	X	X	0	0	X	countQ-;
MulAccH_RV	X	0	1	1	X	X	count	0	0	X	
SaveToAcc	X	0	1	1	X	X	count	1	0	X	countQ=count+1;
SaveToRv	0	0	X	X	X	X	X	0	1	X	count++; en = 0 ?
Result	X	X	X	X	X	X	X	0	0	X	Result : Init

Tabulka 6.1: Kontrolór násobiaceho integrátora v FX.

Stav	SELy0	SEL1	SEL2	SEL3	SELQ	SELR	SELH	RWAcc	RWrv	BLOK	
Reset	0	0	0	0	0	0	0	0	0	0	
Init	1	X	X	X	X	X	X	0	1	X	count = 0; countQ = 0;
MulDQxDRx1 MulDQxDRx2	X	1	0	0	countQ	count-countQ	X	0	0	countQ=count ? 1 : 0	
MulDQxDRx	X	1	0	0	countQ	count-countQ	X	0	0	countQ=count ? 1 : 0	count < N ? WriteToAcc1 : Result
WriteToACC1	X	1	0	0	countQ	count-countQ	X	0	0	countQ=count ? 1 : 0	
WriteToACC	X	1	0	0	countQ	count-countQ	X	1	0	countQ=count ? 1 : 0	countQ = 0 ? MulAccH_RV1 : SetupCounter
SetupCounter	X	X	X	X	X	X	X	0	0	X	countQ-;
MulACCch_RV1 MulACCch_RV2 MulACCch_RV	X	0	1	1	X	X	count	0	0	X	
SaveToAcc1	X	0	1	1	X	X	count	0	0	X	
SaveToAcc	X	0	1	1	X	X	count	1	0	X	countQ=count+1;
SaveToRV	0	0	X	X	X	X	X	0	1	X	count++; en = 0 ?
Result	X	X	X	X	X	X	X	X	X	X	Result : Init

Tabulka 6.2: Kontrolór násobiaceho integrátora v FP.

Stav	SEL2	SEL3	SEL4	SEL5	SEL6	SEL7	SEL8	ClkAcc	ClkY	BLOK	
Reset	000	00	000	000	00	0	0	0	0	0	count = 1; const_int = 1;
Init	000	00	X	000	01	0	X	0	1	1	
MulUH	000	00	X	000	01	0	X	0	0	1	
WaitDiv	000	X	X	X	00	X	X	0	0	X	done = 1 ? SaveDiv : WaitDiv;
SaveDiv	000	X	X	001	00	1	0	0	0	0	
CalculationDY1	000	X	X	001	00	1	0	0	1	0	
WriteToAcc	001	X	X	const_int+1	const_int > 0 ? 10 : 00;	X	X	0	0	1	RWrv = 1; count < (N-1) ? SetupCounters : Result;
SetupCounters	001	X	X	X	const_int > 0 ? 10 : 00;	0	X	0	0	1	count = 1; const_int > 0 ? MulCountDYx : MulDYxDVx;
MulCountDYx	X	X	X	const_int	10	0	X	0	1	1	
SetupDYxDVx	count	X	X	const_int-count+1	00	0	1	0	0	count = 1 ? 1 : 0;	
MulDYxDVx	count	X	X	const_int-count+1	00	0	1	1	0	count < const_int ? SetupCount : SetupDUxH;	
SetupCount	X	X	X	X	X	X	X	0	0	X	count++;
SetupDUxH	X	const_int	X	000	01	0	1	0	0	0	XOR = 1;
MulDUxH	X	const_int	X	000	01	0	1	1	0	0	XOR = 1;
SetupACCdivX	X	X	const_int-1	X	11	1	X	0	0	1	
MulACCdivX	X	X	const_int-1	X	11	1	X	1	0	1	
SetupCalcDYx	000	X	X	const_int+1	00	1	0	0	0	0	
CalculationDYx	000	X	X	const_int+1	00	1	0	1	1	0	
Result	X		X	X	X	X	X	0	0	X	en = 0 ? Result : Init;

Tabulka 6.3: Kontrolór deliaceho integrátora v FX.

Stav	SEL2	SEL3	SEL4	SEL5	SEL6	SEL7	SEL8	ClkAcc	ClkY	BLOK	
Reset	000	00	000	000	00	0	0	0	0	0	count = 1; const_int = 1;
Init	000	00	X	000	01	0	X	0	1	1	
MulUH	000	00	X	000	01	0	X	0	0	1	
MulUH1	000	00	X	000	01	0	X	0	0	1	
SaveDiv	000	X	X	001	00	1	0	0	0	0	ClkV = 1;
SaveAcc	000	X	X	001	00	1	0	1	0	0	
CalculationDY1	000	X	X	001	00	1	0	0	1	0	
CalculationDY2	00	X	X	001	00	1	0	0	0	0	
CalculationDY3	00	X	X	001	00	1	0	1	0	0	
WriteToAcc	001	X	X	const_int+1	const_int > 0 ? 10 : 00;	X	X	0	0	1	RWrV = 1; count < (N-1) ? SetupCounters : Result;
SetupCounters	001	X	X	X	const_int > 0 ? 10 : 00;	0	X	0	0	1	count = 1; const_int > 0 ? MulCountDYx : MulDYxDVx;
SetupCountDYx	X	X	X	const_int	10	0	X	0	0	1	
MulCountDYx	X	X	X	const_int	10	0	X	0	1	1	
SaveCountDYx	X	X	X	const_int	10	0	X	0	1	1	
SetupDYxDVx	X	X	X	const_int	10	0	X	0	1	1	
SetupDYxDVx1	count	X	X	const_int-count+1	00	0	1	0	0	count = 1 ? 1 : 0;	
MulDYxDVx	count	X	X	const_int-count+1	00	0	1	1	0	count = 1 ? 1 : 0;	
SaveDYxDVx	X	X	X	X	X	X	X	0	0	X	count < const_int ? SetupCount : SetupDUxH; count++;
SetupCount	X	X	X	X	X	X	X	0	0	X	
SetupDUxH	X	const_int	X	000	01	0	1	0	0	0	XOR = 1;
MulDUxH	X	const_int	X	000	01	0	1	0	0	0	
SumMulAcc	X	const_int	X	000	01	0	1	0	0	0	
SaveMulAcc	X	const_int	X	000	01	0	1	0	0	0	
SetupACCDivX	X	X	const_int-1	X	11	1	X	1	0	1	
MulACCDivX	X	X	const_int-1	X	11	1	X	0	0	1	
MulACCDivX1	X	X	const_int-1	X	11	1	X	0	0	1	
SumACCDivX	X	X	const_int-1	X	11	1	X	1	0	1	
SaveACCDivX	X	X	const_int-1	X	11	1	X	1	0	1	
SetupCalcDYx	000	X	X	const_int+1	00	1	0	0	0	0	
SetupCalcDYx1	000	X	X	const_int+1	00	1	0	1	1	0	
CalculationDYx	X	X	X	X	X	X	X	0	0	X	en = 0 ? Result : Init;

Tabuľka 6.4: Kontrolór deliaceho integrátora v FP.

## 6.2 Integrátory v pohyblivej rádovej čiarke

Integrátory v pohyblivej rádovej čiarke sú implementované vo verzii bez zdieľaného komponentu *EXP*. Delička *DIV*, násobička *MUL* a sčítačka *SUM* spracúvajú znamienko *S*, exponent *E* a mantisu *M* vnútri seba. Konkrétna implementácia týchto komponentov bola prevzatá a upravená z [1]. Komponenty vykonávajú operáciu násobenia v troch taktoch a operáciu sčítania v dvoch taktoch, keďže pracujeme v pohyblivej rádovej čiarke a výpočet je náročnejší. Podrobný popis priebehu týchto operácií v pohyblivej rádovej čiarke sa nachádza v kapitole 4. Výpočet je teda oproti variantu pracujúcemu s pevnou rádovou čiarkou pomalší, ale o to presnejší. Zároveň sa výrazne zväčšuje rozsah zobraziteľných hodnôt a hustota zobrazenia.

### Jednovstupový integrátor

Jednovstupový integrátor bol v pohyblivej rádovej čiarke implementovaný v 32- a v 64-bitovej verzii. Počet cyklov výpočtu je závislý len od počtu registrov v sete  $h/N$ . Na riadenie integrátora sú rovnako ako v predchádzajúcich variantoch použité signály *RESET* a *EN*. Ostatné signály a stavy konečného automatu sú v kontrolóri zobrazené v tabuľke 6.5. Pri výpočte je použitý jeden čítač. Ten slúži na počítanie jednotlivých cyklov výpočtu a zároveň ako signál na výber hodnoty zo setu registrov  $h/N$ . Výpočet je naštartovaný signálom *Reset*. V ňom sa uložia počiatočné podmienky do registrov *RD* a *RV*. Následne sa pokračuje nasledujúcimi stavmi *Init*, *MulDYxH*, *SetupToRD*, v ktorých prebieha násobenie a začiatok výpočtu sčítania. V stave *SaveToRD* je vypočítaná hodnota nového člena Taylorovej rady, uložená do registra *RD*. V ďalšom stave *SaveToRV* je už hodnota člena sčítaná s hodnotou *RV* a naspäť uložená do registra *RV*. Výpočet  $N$ -krát cyklí a prejde do stavu *Result*. V registri *RV* sa nachádza výsledná hodnota rovnice 3.1.

Stav	SELy0	SEL	SELh	RWrđ	RWrv	
Reset	1	0	0	1	1	
Init	X	X	count	0	0	
MulDYxH	X	X	count	0	0	
SetupToRD	0	1	count	0	0	
SaveToRD	0	1	count	1	0	count++
SaveToRV	X	X	count	0	1	count < N ? Init : Result
Result	X	X	X	0	0	en = 0 ? Result : Init

Tabuľka 6.5: Kontrolór jednovstupového integrátora v FP.

### Deliaci a násobiaci integrátor

Implementovaný bol aj násobiaci a deliaci integrátor v pohyblivej rádovej čiarke a taktiež oba integrátory vo verziách s 32 bitmi a so 64 bitmi. Jednotlivé stavy a nastavenia signálov násobiaceho integrátora sú popísané v tabuľke 6.2. Pri porovnaní s tabuľkou násobiaceho integrátora v pevnej rádovej čiarke 6.1 môžeme vidieť, že výpočet oboch integrátorov prebieha rovnako. Rozdiel je v dĺžke vykonávania operácie násobenia, ktorá potrebuje o dva hodinové takty viac, než variant s FX aritmetikou. Operácia sčítania potrebuje o jeden takt viac.



Popis kontrolóru deliaceho integrátora je v tabuľke 6.4. Pri porovnaní s tabuľkou deliaceho integrátora vo FX aritmetike 6.3 je vidieť, že výpočet prebieha podobne v oboch typoch integrátorov. Rozdiel je v dĺžke vykonávania operácií delenia, násobenia a sčítania, rovnako ako pri násobiacich integrátoroch. Násobenie sa vykonáva v troch taktov hodinového signálu a sčítanie v dvoch taktov hodinového signálu (rovnako ako pri násobiacom integrátore). Zaujímavosťou však je operácia delenia. Vo FP aritmetike je použitá paralelná delička, používajúca paralelný algoritmus bez návratu k nezápornému zvyšku. Dĺžka výpočtu trvá iba jeden hodinový takt.

### 6.3 Sústava integrátorov

Podľa schémy zapojenia 5.8 bol v sekcii 5.7 implementovaný komponent počítajúci diferenciálnu rovnicu 5.1. Komponent tvorí násobiaci a deliaci integrátor v pohyblivej rádovej čiarke so 64 bitmi. Ďalej bol použitý komponent invertor. Každý integrátor v zapojení používa set registrov  $h/N$ . Pre zníženie priestorovej náročnosti bol použitý len jeden set týchto registrov. Veľkosť setu závisí od rádu metódy, ktorú chceme použiť. Výpočet bol implementovaný v dvoch variantoch počítajúcich s Taylorovou metódou 8. rádu a s Taylorovou metódou dokonca až 32. rádu. Každý integrátor je riadený svojím vlastným kontrolórom. Celé zapojenie je ešte ovládané hlavným kontrolórom, ktorý ovláda ostatné, a to pomocou signálov *Reset* a *EN*. Výpočet prebieha v každom integrátore samostatne, a to spôsobom takmer takým istým, aký bol popísaný v predchádzajúcich sekciách tejto kapitoly. Rozdiel je len v tom, že integrátory nepracujú úplne ako samostatné komponenty s predpočítanými hodnotami, ale pracujú v zapojení do sústavy integrátorov, do ktorých vstupné hodnoty prichádzajú postupne. Upravené boli niektoré kroky kontrolóra pri použitých integrátoroch. Upravený kontrolór jednovstupového integrátora je v tabuľke 6.6. Rozdiel je, že integrátor neobsahuje signál *SELh*, keďže set registrov bol vytvorený ako zdieľaný komponent medzi všetkými integrátormi. Signál *SELh* je teraz ovládaný pomocou hlavného kontrolóra. Integrátor neobsahuje ani podmienku na ukončenie výpočtu po dosiahnutí stanoveného počtu cyklov. Tá je aj v tomto prípade v rézii hlavného kontrolóra. Rovnakými úpravami prešiel aj kontrolór násobiaceho integrátora. V násobiacom integrátore bol pridaný ešte jeden výstup na prečítanie výslednej hodnoty z registra *RV*.

Stav	SELy0	SEL	RWrđ	RWrđ
<b>Reset</b>	1	0	1	0
<b>Init</b>	X	X	0	1
<b>MulDYxH</b>	X	X	0	0
<b>SetupToRD</b>	0	1	0	0
<b>SaveToRD</b>	0	1	1	1
<b>Result</b>	X	X	0	0
	en = 0 ? Result : Init			

Tabuľka 6.6: Kontrolór jednovstupového integrátora v FP v zapojení do sústavy.

Na začiatku výpočtu je hlavný kontrolór uvedený do stavu *Reset* a signálom privedie ostatné integrátory tiež do stavu *Reset*. V tomto stave sú nahrané počiatkové podmienky do integrátorov. Tie sú privedené na výstup každého integrátora. Hodnota sa tak dostane na potrebné vstupy integrátorov. Následne je pomocou povolovacieho signálu *EN* spustený vý-

počet na všetkých integrátoroch naraz. Vstupné hodnoty násobiaceho integrátora sú uložené do setu registrov  $DR(N-1)$  a  $DQ(N-1)$ . Výpočet pokračuje samostatne v každom integrátore. Keďže však výpočet násobiaceho integrátora trvá najdlhšie z použitých integrátorov, ostatné integrátory  $q$ ,  $r$  a  $s$  po dokončení výpočtu čakajú na násobiaci integrátor. Ten po dokončení výpočtu aktivuje signál *OEN* (*output enable*) na jednotku. Tento signál je privedený do hlavného kontrolóru, ktorý následne spustí ďalší cyklus výpočtu nastavením signálu *EN* do 1 v každom integrátore schémy. Toto sa opakuje až kým výpočet jedného kroku metódy neskončí. Pri výpočte ďalšieho kroku výpočtu sú obnovené počiatočné podmienky a výpočet sa znova vykoná.

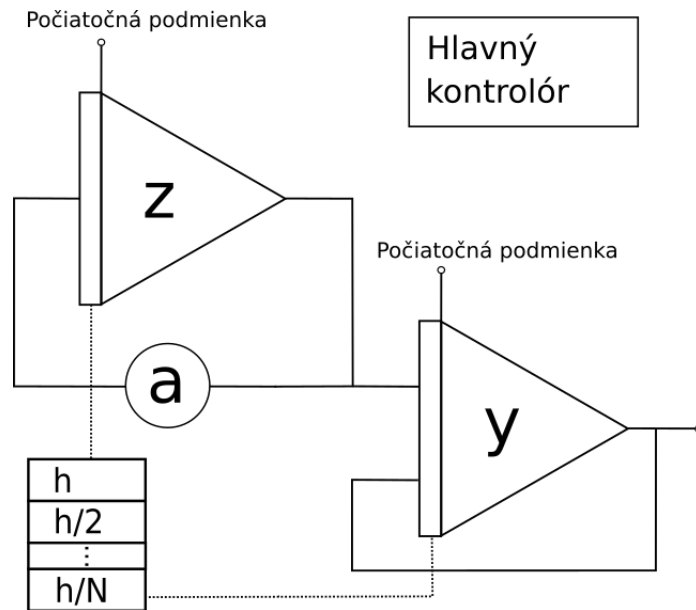
FPGA na prípravku FitKit 2.0 nie je dostatočne veľké na implementáciu tejto sústavy rovníc, predstavenej v sekcii 5.7, preto bola navrhnutá jednoduchšia diferenciálna rovnica v tvare

$$y' = y * e^{at} \quad (6.1)$$

Rovnicu môžeme prepísať do sústavy diferenciálnych rovníc:

$$\begin{aligned} y' &= y * z, & y(0) &= 1 \\ z' &= a * z, & z(0) &= 1, \text{ kde } a < 0 \end{aligned} \quad (6.2)$$

Zo sústavy rovníc vytvoríme schému zapojenia integrátorov, ktorá je na obrázku 6.4. Obsahuje len dva integrátory, jeden jednovstupový integrátor a jeden násobiaci integrátor, oba v pevnej rádovej čiarke na 32 bitoch. Pre zníženie miesta je rovnica počítaná pomocou Taylorovej metódy 4. rádu. Komponent  $a$  predstavuje konštantu, ktorou je násobená výstupná hodnota integrátora  $z$ . Vynásobená hodnota je privedená naspäť na vstup tohto integrátora. Toto zapojenie počíta funkciu  $e^{at}$ . Integrátor  $y$  je taktiež zapojený do cyklu a spolu s integrátorom  $z$  a s komponentom  $a$  tvoria schému zapojenia, počítajúc zadanú rovnicu. Do oboch integrátorov je privedená zbernica zo setu registrov  $h$ .



Obr. 6.4: Schéma zapojenia integrátorov.

Ovládanie komponentov na prípravku FitKit 2.0 je realizované cez terminál programu QDevKit. Komunikácia medzi MCU (Micro-Controller Unit) a FPGA prebieha pomocou rozhrania *SPI*. Jednotlivé hodnoty zadávané v termináli sú ukladané do pamäti a cez prevodník *SPI\_ADC* privedené do registrov FPGA. Jednotlivé registre sú vybrané adresovým dekóderom. Po inicializácii komunikácie prebieha v nekonečnom cykle metódou pooling čítanie z riadiaceho registra a z registra výsledku. Hodnoty registrov sú vypisované v termináli v hexadecimálnej forme. Príkazom *help* je možné zobrazíť nápovedu ovládania integrátora. Príkazmi *DATAY* a *DATAV* je možné nahráť počiatočné podmienky do integrátorov *y* a *v*. Hodnoty počiatočných podmienok je potrebné zadávať v hexadeximálnej forme s veľkými písmenami, keďže sa vykonáva prevod z textovej formy do číselnej, ktorá je následne uložená do registra. Krok metódy je stanovený na hodnotu *0.1* a nie je možné meniť ho počas priebehu aplikácie. To je z dôvodu potreby predpočítania podielov kroku *h*. Tie sú už predpočítané v sete registrov. Po zadaní počiatočných podmienok je možné spustiť výpočet pomocou príkazu *START*. Tým sa nastaví povolovací signál *EN* a začne sa výpočet. Následne je výsledok po jednom kroku zobrazený v termináli. Pre výpočet hodnoty v ďalšom kroku je potrebné zadať získaný výsledok ako počiatočnú podmienku násobiaceho integrátora *y*. Novú počiatočnú podmienku je potrebné zadať aj pri jednovstupovom integrátore, tú však nepoznáme. Jednoduchým a užívateľsky príjemným riešením je automatické kopírovanie výsledných hodnôt v oboch integrátoroch z registrov *RV* do registrov *RD*. Preto bol pridaný signál *RvToRd* do jednovstupového integrátora, ktorý umožní presun z jedného registra do druhého bez potreby nahrania počiatočnej podmienky príkazom *DATAV*. Presun hodnoty v násobiacom integrátore je realizovaný na úrovni MCU, pomocou ktorého sa hodnota získaného výsledku uloží ako počiatočná podmienka. Celý tento postup je skrytý za jednoduchým príkazom *NEXTY*.



(a) VIRTEX-5 s FPGA XC5VSX50T



(b) FitKitu 2.0 s FPGA Spartan 3 XC3S400

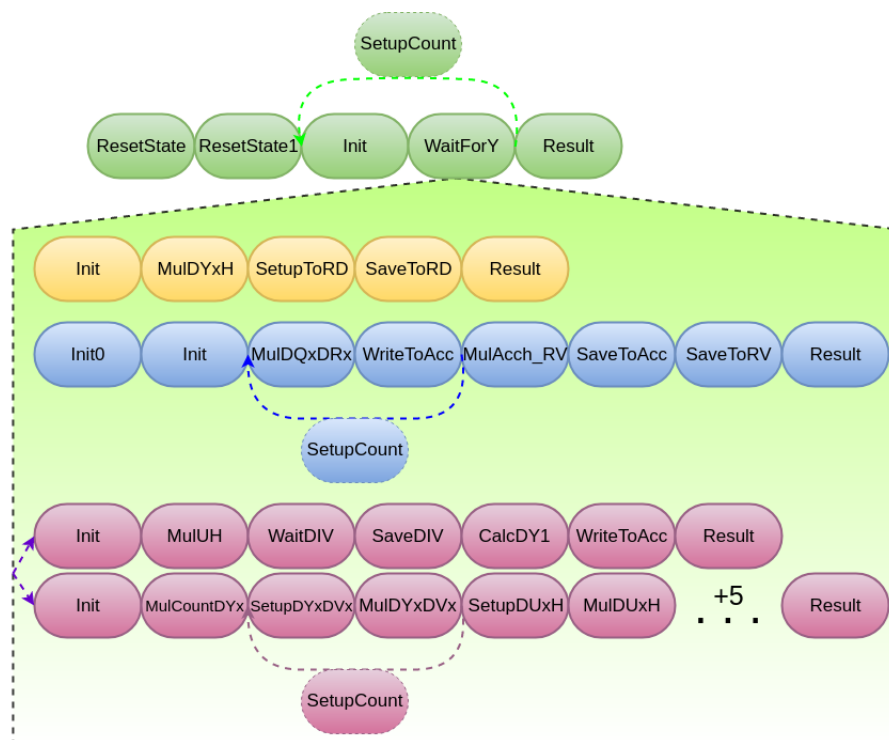
Obr. 6.5: Prípravky FitKit 2.0 a VIRTEX-5, s ktorými sa pracovalo.

# Kapitola 7

## Analýza

V tejto kapitole zanalyzujeme dosiahnuté výsledky. Analyzovať môžeme rôzne vlastnosti navrhnutého procesora a taktiež použitej Taylorovej rady. Najdôležitejšími vlastnosťami sú presnosť výsledku, časová náročnosť a priestorová zložitosť. V tejto analýze sa budeme zaoberať časovou zložitosťou numerických metód a ich presnosťou. Porovnáme Taylorovu radu s metódami nižších rádov, ako sú Eulerova metóda, metóda Runge-Kutta 2. rádu a metóda Runge-Kutta 4. rádu. Tieto metódy sú popísané v kapitole 2.

V prvom rade vypočítame čas procesora, potrebný na výpočet sústavy rovníc 6.2. V každom takte sa procesor nachádza v nejakom stave. Z výpisu jednotlivých stavov kontrolórov tak môžeme vypočítať, koľko stavov je potrebných na výpočet pri použití rôzneho rádu metódy. Výsledný počet stavov potom uvádza, koľko taktov procesora je potrebných na výpočet. Z frekvencie procesora a z počtu taktov následne získame celkový čas výpočtu.



Obr. 7.1: Stavy jednotlivých integrátorov.

Zobrazenie jednotlivých stavov kontrolórov je na obrázku 7.1. Jednotlivé bunky predstavujú stavy kontrolórov. Zelenou farbou je znázornený hlavný kontrolór, ktorý ovláda oba integrátory. Žltou farbou sú znázornené stavy jednovstupového integrátora a modrou farbou stavy násobiaceho integrátora. Hlavný kontrolór je v stave *WaitForY* počas celého výpočtu integrátorov. Z tohto stavu prejde do ďalšieho stavu až po skončení výpočtu násobiaceho integrátora, keďže jeho výpočet trvá najdlhšie. Ak bol dosiahnutý maximálny rád metódy, hlavný kontrolór prejde do stavu *Result*. V opačnom prípade výpočet pokračuje stavom *SetupCount* a začne sa nový cyklus výpočtu. Počet taktov  $T_h$  hlavného kontrolóra, a teda celého výpočtu, môžeme vyjadriť vzťahom

$$\begin{aligned} T_h &= 2 + 1 + W + 2(N - 1) + 1 \\ T_h &= 2 + W + 2N, \end{aligned} \quad (7.1)$$

kde  $N$  označuje počet cyklov výpočtu, čiže použitý rád Taylorovej metódy. Symbol  $W$  znamená počet stavov (taktov) v stave *WaitForY*. Dĺžka tohto stavu sa rovná počtu stavov najdlhšie trvajúceho komponentu v sústave zapojenia. Označme potom  $W$  ako  $W_{jv}$  pre jednovstupový integrátor,  $W_{nas}$  pre násobiaci integrátor a  $W_{div}$  pre deliaci integrátor. Ak je v zapojení len jednovstupový integrátor, počet stavov je 5. Celkový čas výpočtu potom vyjadríme nasledovne:

$$\begin{aligned} W_{jv} &= 5N \\ T_{jv} &= (2 + W_{jv} + 2N) * t_{clk} \\ T_{jv} &= (2 + 5N + 2N) * t_{clk} \\ T_{jv} &= 2t_{clk} + 7t_{clk}N. \end{aligned} \quad (7.2)$$

Symbol  $t_{clk}$  udáva čas periódy jedného taktu. Podobný výpočet vykonáme aj pre násobiaci a pre deliaci integrátor. Výpočet násobiaceho integrátora trvá 8 stavov a v každom ďalšom cykle je ich počet zvýšený o štyri. Z toho vyplýva, že

$$\begin{aligned} W_{nas} &= 6N + N(N + 1) + \frac{N(N - 1)}{2} \\ T_{nas} &= \left( 2 + W_{nas} + 2N \right) * t_{clk} \\ T_{nas} &= \left( 2 + 6N + N(N + 1) + \frac{N(N - 1)}{2} + 2N \right) * t_{clk} \\ T_{nas} &= \left( 2 + 9N + N^2 + \frac{N(N - 1)}{2} \right) * t_{clk}. \end{aligned} \quad (7.3)$$

Výpočet počtu krokov deliaceho integrátora je zložitejší. V prvom cykle sa vykonáva delenie a až pri výpočte ďalších členov Taylorovej rady sa násobia predchádzajúce členy  $DVx$  a  $DUx$  vo všetkých kombináciách medzi sebou. Výpočet stavov rozdelíme na počet stavov pre výpočet prvého člena a počet stavov pre výpočet ostatných členov Taylorovej rady. Rovnaké rozdelenie môžeme vidieť aj v konečnom automate deliaceho integrátora

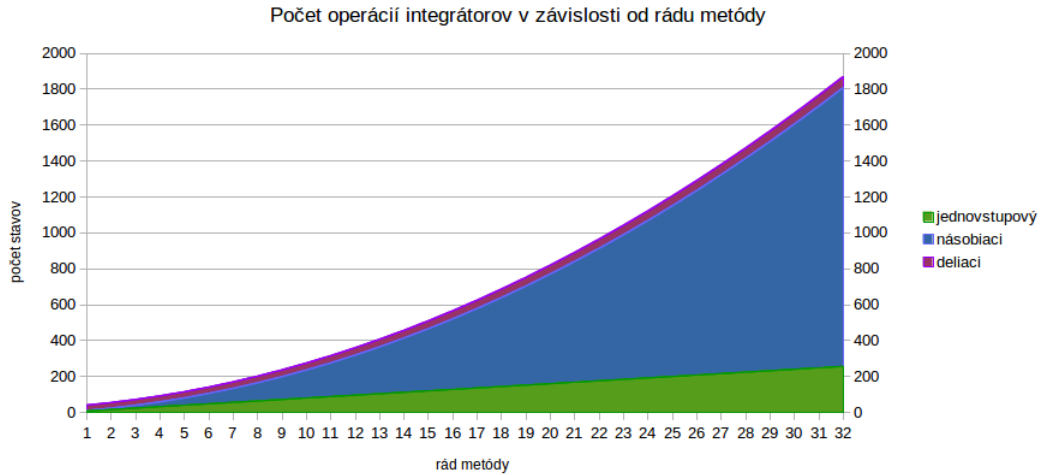
z obrázku 7.1. Počet stavov  $W_{div_1}$  pre prvý člen Taylorovej rady, počet stavov ostatných členov  $W_{div_x}$  a celkový počet krokov deliaceho integrátora vypočítame nasledovne:

$$\begin{aligned}
 W_{div_1} &= 6 + DIV \\
 W_{div_x} &= 10(N-1) + N(N-1) + \frac{(N-1)(N-2)}{2} \\
 W_{div} &= W_{div_1} + W_{div_x} \\
 W_{div} &= 9N - 4 + DIV + N^2 + \frac{N^2 - 3N + 2}{2}.
 \end{aligned} \tag{7.4}$$

Celkový čas výpočtu je potom

$$\begin{aligned}
 T_{div} &= (2 + W_{div} + 2N) * t_{clk} \\
 T_{div} &= \left( 2 + 9N - 4 + DIV + N^2 + \frac{N^2 - 3N + 2}{2} + 2N \right) * t_{clk} \\
 T_{div} &= \left( DIV - 2 + 11N + N^2 + \frac{N^2 - 3N + 2}{2} \right) * t_{clk}.
 \end{aligned} \tag{7.5}$$

Zo vzťahov môžeme vidieť, že časová zložitosť výpočtu pri jednovstupovom integrátore je lineárna, pri násobiacom a pri deliacom integrátore je kvadratická. Časová zložitosť je znázornená na obrázku 7.2. Je vidieť, že počet operácií deliaceho integrátora je len o niečo väčší než pri násobiacom integrátore.



Obr. 7.2: Schéma zapojenia integrátorov.

Ďalším analyzovaným kritériom je presnosť numerických metód. Na porovnanie presnosti potrebujeme poznať exaktné riešenie problému, ktorý riešime. Rovnica (6.1) má exaktné riešenie nasledovné:

$$y' = e^{\frac{at}{a}} e^{-\frac{1}{a}}. \quad (7.6)$$

Porovnanie jednotlivých metód je zobrazené v tabuľkách 7.1 a 7.2. Porovnanie jednotlivých metód spočívalo v nájdení maximálneho kroku  $h_{max}$  každej z metód tak, aby rozdiel najvyšších členov metód v čase  $t_{max}$  bol čo najmenší. Chyba výpočtu Eulerovej metódy bola určená podľa vzťahu:

$$error = |y_{euler}(t_{max}) - y_{exact}(t_{max})|. \quad (7.7)$$

Rovnaký postup výpočtu chyby bol použitý aj pri ostatných metódach. Výsledný počet operácií (stavov) výpočtu a chybu výsledku nájdeme v tabuľkách 7.1 a 7.2. Tabuľka 7.1 bola vytvorená s krokom  $h = 1.0$  pre Taylorovu metódu 8. rádu. Maximálny čas bol stanovený na  $h_{max} = 1.0$ . Z tabuľky je vidieť, že najmenšiu chybu mala Eulerova metóda, čo je však spôsobené veľmi malým krokom, ktorý viedol k obrovskému množstvu operácií. Najlepší pomer medzi presnosťou a rýchlosťou majú metódy Runge-Kutta 4. rádu a Taylorova metóda. Chyba Taylorovej metódy je spôsobená hlavne pre veľkosť kroku.

metóda	$h_{max}$	hodnota najvyššieho člena	$t_{max}$	výsledok	error	počet krokov [ $t_{max}/h_{max}$ ]	počet operácií [ $T * (t_{max}/h_{max})$ ]
Euler	0.00044	3.045E-04	1.00012	1.88165390491486	2.554E-05	2273	27 276
R-K 2	0.03750	-3.075E-04	1.01250	1.89048695564270	2.722E-04	27	675
R-K 4.	0.35000	3.177E-04	1.05000	1.91570568084717	4.560E-05	3	123
Taylor 8	1.00000	-3.063E-04	1.00000	1.88219246082008	5.961E-04	1	60

Tabuľka 7.1: Porovnanie presnosti a rýchlosti numerických metód.

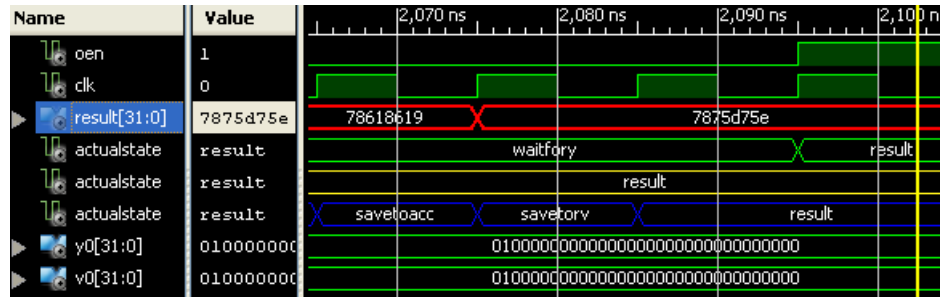
V tabuľke 7.1 bol numerický krok Taylorovej metódy 8. rádu znížený na polovicu, t. j.  $h = 0.5$ . Eulerovu metódu už neberieme do úvahy, keďže počet operácií by bol priveľký. Pri porovnaní oboch tabuliek je vidieť, že pomer medzi presnosťou a rýchlosťou je najlepší pri Taylorovej metóde 8. rádu. Aby sme zachovali rovnakú presnosť Taylorovej metódy 8. rádu, v metóde Runge-Kutta 4. rádu je potrebné zvoliť dostatočne malý krok, avšak počet operácií sa tým niekoľkokrát zvýši. V tomto konkrétnom príklade sa počet operácií zvýšil štvornásobne. Ďalej si môžeme všimnúť, že metóda si udržiava presnosť vzhľadom na hodnotu posledného člena metódy.

metóda	$h_{max}$	hodnota najvyššieho člena	$t_{max}$	výsledok	error	počet krokov [ $t_{max}/h_{max}$ ]	počet operácií [ $T * (t_{max}/h_{max})$ ]
R-K 2	0.00234	-1.198E-06	0.99918	1.88102829456329	2.377E-06	427	10 675
R-K 4.	0.08580	1.197E-06	1.02960	1.90189445018768	4.788E-07	12	492
Taylor 8	0.50000	1.196E-06	1.00000	1.88159812241793	1.735E-06	1	120

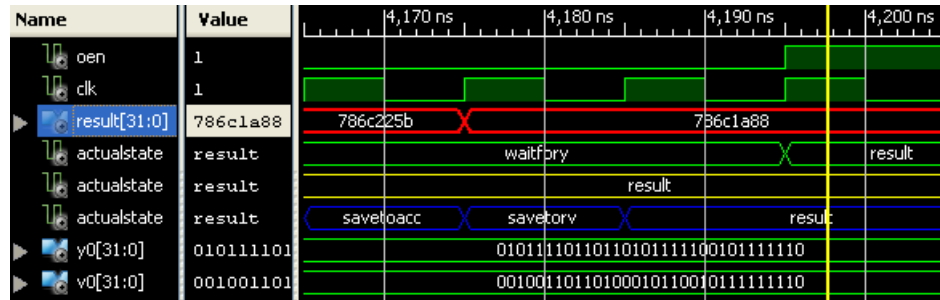
Tabuľka 7.2: Porovnanie presnosti a rýchlosti numerických metód.



Nasledujúce obrázky 7.3 a 7.4 zachytávajú simuláciu Taylorovej rady 8. rádu v programe ModelSim na výpočet hodnôt v predchádzajúcich tabuľkách. Na obrázkoch je červenou farbou zvýraznený register výsledku  $RV$  násobiaceho integrátora. Stav označený zelenou farbou patrí hlavnému kontrolóru. Žltou farbou je označený aktuálny stav jednovstupového integrátora, modrou farbou sú označené stavy násobiaceho integrátora. Hodnota v registri je zobrazená hexadecimálne a zodpovedá decimálnej hodnote uvedenej ako výsledok v predchádzajúcich tabuľkách.



Obr. 7.3: Simulácia sústavy rovníc 6.2 Taylorovou radou 8. rádu s  $h = 1$  a  $h_{max} = 1$ .



Obr. 7.4: Simulácia sústavy rovníc 6.2 Taylorovou radou 8. rádu s  $h = 0.5$  a  $h_{max} = 1$ .



## Kapitola 8

# Záver

V tejto práci sme sa zaoberali riešením počiatočných úloh s využitím numerickej integrácie pomocou modernej metódy Taylorovej rady. Úpravou jej členov sme získali potrebné rovnice, z ktorých boli vytvorené jednotlivé návrhy integrátorov. Navrhli sme paralelné integrátory v pevnej a v pohyblivej rádovej čiarky s operáciou násobenia a delenia. Ďalej sme rozšírili návrhy týchto paralelných integrátorov tak, aby bol nimi možný výpočet ako samostatného komponentu, tak aj komponentu zapojeného do sústavy. Následne sme navrhnuté integrátory popísali vo VHDL a otestovali ich pomocou *testbenchu* v simulačnom nástroji *ModelSim*. Takto pripravené integrátory sme potom implementovali v FPGA, kde bola otestovaná ich funkčnosť. Implementovaný bol aj jednovstupový integrátor navrhnutý v práci [7]. Tento paralelný integrátor bol taktiež implementovaný v prevedení pevnej a pohyblivej rádovej čiarky. Na ovládanie integrátorov sme použili terminál programu *QDevKit*. Zjednodušili sme ovládanie integrátorov pridaním príkazu *nexty*, ktorý umožňuje výpočet ďalšieho kroku riešenej rovnice bez opätovného zadávania počiatočných podmienok.

Takto implementované integrátory môžu byť použité pri výučbe hardvérových predmetov a predmetov venujúcich sa numerickej integrácii. V poslednej kapitole sme sa venovali časovej analýze navrhnutých integrátorov. Ďalej sme porovnali presnosť Taylorovej metódy s Eulerovou metódou a s metódami Runge-Kutta 2. a 4. rádu. Z analýzy sme zistili, že najlepší pomer medzi presnosťou a rýchlosťou z porovnávaných metód má Taylorova rada 8. rádu.

Ďalším pokračovaním práce môže byť návrh a implementácia kontrolóra pre rozsiahlejšie rovnice s použitím väčšieho programovateľného hradlového poľa. Rozsiahlejšou témou by mohlo byť vytvorenie prepojovacej siete integrátorov s dynamickou konfiguráciou prepojení pre riešenie rôznych sústav diferenciálnych rovníc.

# Literatúra

- [1] Jean Pierre Deschamps, Gustavo D. Sutter, Enrique Cantó: *Guide to FPGA Implementation of Arithmetic Functions*. Springer, 2012, ISBN 978-94-007-2986-5, VHDL kódy dostupné z:  
[http://www.arithmetic-circuits.org/guide2fpga/vhdl\\_codes.htm](http://www.arithmetic-circuits.org/guide2fpga/vhdl_codes.htm).
- [2] Kraus, M.: *Paralelní výpočetní architektury založené na numerické integraci*. Disertační práce, FIT VUT v Brně, 2013.
- [3] Kunovský, J.: *Modern Taylor Series Method*. Habilitation work, VUT Brno, 1994.
- [4] Matečný, F.: *Simulátor procesora s operáciou delenia*. Bakalářská práce, FIT VUT v Brně, 2016.
- [5] Milan Kubíček, D. J., Miroslava Dubcová: *Numerické metody a algoritmy*. VŠCHT Praha, 2005, ISBN 80-7080-558-7.
- [6] Opálka, J.: *Automatické řízení výpočtu*. Bakalářská práce, FIT VUT v Brně, 2014.
- [7] Opálka, J.: *Automatické řízení výpočtu ve specializovaném výpočetním systému*. Diplomová práce, FIT VUT v Brně, 2016.
- [8] Sekanina, L.: Operace v FP a iterační algoritmy. , 2017, slajdy k predmetu INP - Návrh počítačových systémů.
- [9] Sekanina, L.: Reprezentace dat. , 2017, slajdy k predmetu INP - Návrh počítačových systémů.
- [10] Tišnovský, P.: Fixed point arithmetic [online].  
<http://www.root.cz/clanky/fixed-point-arithmetic/>, 2006-05-24 [cit. 2017-11-22].
- [11] Tišnovský, P.: Norma IEEE 754 a příbuzní: formáty plovoucí řádové tečky [online].  
<https://www.root.cz/clanky/norma-ieee-754-a-pribuzni-formaty-plovouci-radove-tecky/>, 2006-05-31 [cit. 2017-11-22].
- [12] Tišnovský, P.: Aritmetické operace s hodnotami ve formátu plovoucí řádové čárky [online].  
<https://www.root.cz/clanky/aritmeticke-operace-s-hodnotami-ve-formatu-plovouci-radove-carky/>, 2006-06-07 [cit. 2018-01-07].
- [13] Vašíček, Z.: FITkit [online]. <http://merlin.fit.vutbr.cz/FITkit/>, 2006 - 2012 [cit. 2018-05-16].

- [14] Závada, V.: *Simulátor procesoru s operací násobení*. Bakalářska práce, FIT VUT v Brně, 2016.
- [15] Čambor, M.: *Elementární procesor v aritmetice pevné a pohyblivé řádové čárky*. Bakalářska práce, FIT VUT v Brně, 2009.
- [16] Šátek, V.: *Analýza stiff soustav diferenciálních rovnic*. Disertační práce, FIT VUT v Brně, 2011.

# Príloha A

## Obsah CD

Priložené CD obsahuje:

- Text tejto práce vo formáte Pdf.
- Zdrojové súbory tejto práce v  $\text{\LaTeX}$ u.
- Zdrojové súbory rôznych typov paralelných integrátorov vo VHDL.
- Popis ovládania integrátorov v prostredí QDevKit.